File No. 7090-25
Form C28-6066-6

IBM Systems Reference Library

IBM 7090/7094 Programming Systems
FORTRAN II Operations

The IBM 7090/7094 FORTRAN II System consists of the FORTRAN II Compiler, the FAP (FORTRAN Assembly Program) Assembler, and the FORTRAN II Monitor. The Monitor coordinates compilation, assembly, and execution of FORTRAN and FAP programs.

This publication provides instructions for the operation and use of the IBM 7090/7094 FORTRAN II System and for the creation and maintenance of the FORTRAN II System tape.

INTRODUCTION

The 7090/7094 FORTRAN II Processor is composed of the FORTRAN II Compiler, the FAP (FORTRAN Assembly Program) Assembler, and the FORTRAN II Monitor. The FORTRAN II Monitor coordinates compilations and assemblies, and permits binary object programs from previous compilations or assemblies to be executed together as parts of a single job. The FORTRAN II Processor operates under the control of the FORTRAN II Monitor which may, in turn, operate under the control of the IBM Basic Monitor (IBSYS). When operating under IBSYS, jobs for the FORTRAN II Monitor may be stacked as input along with jobs for other processors that operate under IBSYS (e.g., the IBJOB Processor).

This publication contains a description of the FORTRAN II Processor operating under the FORTRAN II Monitor (Part 1 of this publication) and of the FORTRAN II Processor operating under IBSYS (Part 2 of this publication).

Knowledge of the contents of the following publications is assumed:

IBM 7090/7094 Programming Systems: FORTRAN II, Form C28-6054-4
IBM 709/7090 Programming Systems: FORTRAN Assembly Program (FAP), Form C28-6235-2
IBM 709/7090 Programming Systems: FORTRAN - Input/Output Package for 32K Version, Form J28-6190
32K 709/7090 FORTRAN: Adding Built-In-Functions, Form J28-6135
IBM 7090/7094 Operating Systems: Basic Monitor (IBSYS), Form C28-6248

This edition, Form C28-6066-6, is a reprint of Form C28-6066-5 incorporating changes released in Technical Newsletters N28-0114-0 and N28-0119-0. The original publication and the newsletters are not obsolete.

CONTENTS

CHAPTER 1: THE 7090/7094 FORTRAN II MONITOR SYSTEM

The 7090/7094 FORTRAN II Monitor System, as distributed, consists of the following items:

1. The Master Tape
2. The Master Tape Duplicating program - 9FTCP
3. The Editor deck
   (Items 1 and 3 are used to produce the FORTRAN II Monitor System Tape.)
4. Test Program - NIM

Each of these items is discussed in the paragraphs which follow.

The Master Tape

The FORTRAN Monitor System is produced by editing the Master tape with the Editor deck. Only the FORTRAN Monitor System tape, thus generated, may be used for source program compilation/assembly and/or object program execution.

The Master tape consists of four files:

File 1 contains both the routines that perform Monitor functions and the FAP assembler.

File 2 contains the FORTRAN Compiler.

File 3 is the library. This file consists of input/output and exponential subroutines utilized by the FORTRAN Compiler, and the FORTRAN library mathematical subroutines (see Chapter 2).

The first record of each library subroutine is a program card record. This record contains the information (length of common storage, number of instructions, etc.) required by the system to incorporate the associated subroutine into an object program. The succeeding records of each library subroutine are the subroutine instructions in relocatable binary form.

File 4 contains the General Diagnostic, which contains routines to analyze certain source program errors and produce a message specifying the error. The General Diagnostic is used only by the FORTRAN Compiler.

The Master Tape Duplicating Program - 9FTCP

In order to maintain the FORTRAN Monitor System intact, it is advisable to use the Master tape as little as possible, and then only to produce copies of the Master tape. Copies produced from the tape are used for the generation of a FORTRAN Monitor System.

The Master Tape Duplicating program is supplied with the FORTRAN System to produce copies of the Master tape for subsequent processing. To insure the accurate duplication of the Master tape, the 9FTCP program performs a checksum verification of all records duplicated from the Master tape.

9FTCP may also be used to copy a System tape.

The Editor Deck

The Editor deck is the means by which a System tape is generated from the Master tape (or more precisely, from a copy of the Master tape). The System tape is defined to be a tape that contains the FORTRAN II Monitor System; it is used for the compilation and assembly of programs.

The Editor deck may also be written onto tape off-line; the Edit tape produced permits a System tape to be generated from the FORTRAN Master tape as a Monitor job. Thus, the Edit tape may be used to update a System tape and to add or delete library subroutines.

Composition of the Editor Deck

The Editor deck is composed of the following column binary and Monitor control cards:

1. Identification card - *I. D.
2. Execute card - *XEQ.
3. Call Edit program - 9CLEDT.
4. 7090/7094 FORTRAN Editing program - 9EDIT.
5. Data card - *DATA.
6. 9FIOP cards, if any.
7. System Edit control card - 9SYSEDIT.
8. Record control cards for those records of the Monitor which are to be changed, if any. Following the record control cards are the binary correction and/or instruction cards for the record. Following all Monitor records is the control card to write an end of file on the System tape after the Monitor.
9. Same as 8, but for the Compiler.
10. Library Edit control card - 9LIBEDIT.
11. Library subroutine program card and relocatable instruction cards for each library subroutine.
12. Diagnostic Edit control card - 9DIGEDIT.
13. Diagnostic record control card followed by the binary correction and/or instruction cards and the control card to write an end of file on the System tape after the Diagnostic.
14. End-of-Edit control card - 9ENDEDIT.

Note that the Editor deck is set up as a Monitor job with the necessary control cards. 9CLEDT is a main program that calls the subroutine 9EDIT. The data for this object program are column binary instruction cards, edit control cards, record cards, and correction and/or change cards that are recognized by 9EDIT.

The output of the above Monitor job is an updated System tape; the Master tape and Edit tape are unchanged.

Maintenance of the Editor Deck

Maintenance of the FORTRAN Editor deck will be by addition or deletion of binary cards containing corrections and additions to the System. These cards are distributed by the IBM DP Program Information Department. Memoranda accompanying these changes or correction cards will indicate how the cards are to be incorporated into the Editor deck, and the reason for the change or correction.

The following types of corrections or changes are possible in the FORTRAN System:

Change Existing Record on the Master Tape: If the contemplated change does not involve changing the first location that the record is to occupy when loaded into core storage, the record word count, or the entry point, it is sufficient to insert the required change card(s) after the last binary card (if any) following the Master Record card. If there is a change in the record's entry point, first location, or word count, then its Master Record card must be changed accordingly (discussed later in this chapter).

If there are no changes for this record currently in the Editor deck, there will be no Master Record card for the record. When it is necessary to alter such a record, a Master Record card followed by the change card(s) must be inserted in the Editor deck.

Delete the Existing Record on the Master Tape: Make the word count zero on the appropriate Master Record card.

Insert a New Record: Insert a New Record card, followed by the required instruction cards, into the Editor deck where the new record is desired.

Add or Delete a File Mark: Insert or remove the corresponding End File card in the Editor deck.

Test Program - NIM

This program is a typical FORTRAN II program which serves as a test deck for the System.

CHAPTER 2: GENERATING AND MAINTAINING A SYSTEM TAPE

The FORTRAN II Master tape and the FORTRAN II System tape each consist of four files of records. All files of the System tape are generated by the FORTRAN II Editing program, 9EDIT, from the corresponding files on the Master tape and from the record control cards and binary instruction cards that are included in the Editor deck.

The method by which the System tape is generated is outlined below in the description of 9EDIT.

The Call Edit Program - 9CLEDT

9CLEDT is a main program that calls the Editing program, 9EDIT, and specifies which tape units contain the new System tape, the Master tape, and the Edit tape. The standard calling program, as distributed, specifies B6 as the Master tape, A6 as the new System tape, and A2 as the Edit tape. In addition, 9CLEDT specifies the tape on which the library is to be written; as distributed, 9CLEDT specifies that the library is to be written on the new System tape. In order to change tape assignments, the calling program must be FAP assembled with the desired parameters, as in the following example:

```
       FAP
       STL*   $FLAGB5 (Include only to generate
                library on B5)
       STL*   $FLAGA1 (Include only to suppress
                library on A1)
       CALL   EDIT, SYSTEM, MASTER, EDITAP
       CALL   EXIT
SYSTEM OCT    1206
MASTER OCT    2206
EDITAP OCT    1202
       END
```

The above program will replace 9CLEDT in the Editor deck.

It is possible to edit from a System tape on logical tape 1, for example, by reassembling 9CLEDT to change symbolic location MASTER to OCT 1201. This will cause the current System tape to be used as the Master tape. Upon completion of editing, it is possible to continue processing using either the old System tape or the newly created System tape by dialing the desired one to logical tape 1.

The FORTRAN Editing Program - 9 EDIT

Each record of the Monitor and Compiler files of the System tape is generated as follows:

1. A Master or New Record card is read from the Edit tape.

2. All records whose numbers are less than that defined by the Master or New Record card are copied from the Master tape onto the System tape.
3. The record that is to be changed, i.e., corresponding to the Master or New Record card, is read into a buffer, corrected, and written onto the System tape. This record must contain fewer than 10,000 words.
4. The limits of the record are determined by the first location and the word count given on a Master or New Record card. The record is considered bounded by these limits.
5. The first two words of each record correspond to the third and fourth words of the Master or New Record card. These words are considered control words.

When 9EDIT recognizes the Library Edit control card, 9LIBEDIT, it proceeds as follows: If the library exists on the Edit tape, it is copied onto the System tape; if there is no library on the Edit tape, the library is copied from the Master tape.

When 9EDIT recognizes the Diagnostic Edit control card, 9DIGEDIT, it will process cards for the Diagnostic file in the same manner as it processed the records of the Monitor and Compiler files.

Even if only one record of the System is being changed, all end-of-file control cards and the 9LIBEDIT and 9DIGEDIT cards must be in the Editor deck.

To change IOP (the Input/Output Package), the first card of IOP, 9FIOP001, and the last card of IOP, 9FIOPU01, with the patch cards between them, must be placed after the * DATA card which immediately precedes the card 9SYSEDIT.

9EDIT contains control cards to write an end of file on the new System tape after record 12, after record 34, and after the Diagnostic, which is the last record on the System tape. These same control cards, along with the necessary Master record cards for the records being corrected, enable editing to be effected from either a Master tape or a System tape.

Cards Recognized by 9EDIT

9EDIT will recognize thirteen types of cards which may follow it in the Editor deck. The formats of these column binary cards are given below. (Note: Record numbers are entered as the octal equivalent of ten times the actual record number.)

(All numbers are octal)

Description	Word	Prefix	Decrement	Tag	Address
1. System Edit control card - 9SYSEDIT	1	6	500 + word count	--	1
2. Master Record card	1	4	500 + word count	--	0
	2	←	Checksum		→
	3	3	Record word count	--	First location
	4	1	Record number	--	Entry point
3. New Record card	1	5	500 + word count	--	0
	2	←	Checksum		→
	3	3	Record word count	--	First location
	4	1	Record number	--	Entry point

Card types 4, 5, 6, and 7 are End-of-File control cards

Description	Word	Prefix	Decrement	Tag	Address
4. Skip an end of file on the Master tape, but write an end of file on the System tape after record number n.	1	7	500	--	1
	2	0	n	--	0
5. Write an end of file on the System tape after record number n.	1	7	500	--	2
	2	0	n	--	0
6. Skip an end of file on the Master tape after record number n.	1	7	500	--	3
	2	0	n	--	0

Description	Word	Prefix	Decrement	Tag	Address
		(All numbers are octal)			
7. Skip an end of file on the Master tape after record number n and block further input from the Master tape until the mode of editing changes. This will enable new records to be added just before the end of file.	1	7	500	--	4
	2	0	n	--	0
8. Library Edit control card - 9LIBEDIT.	1	6	500	--	2
9. Diagnostic Edit control card - 9DIGEDIT.	1	6	500	--	3
10. End-of-Edit control card - 9ENDEDIT.	1	6	500	--	4
11. Instruction card (22 words per card).	1	0	500 plus word count	--	load address
	2	← Checksum →			
	3	← First binary correction instruction →			
	4	← Second binary correction instruction →			
	⋮				

	Word	Bit S	1-5	6-8	9-11	12-23	24-35
12. Instruction card (23 words per card).	1		Word count	High-order digit of load address	5	Remainder of load address	Folded checksum
	2	← First binary correction instruction →					
	3	← Secondary binary correction instruction →					
	⋮						

13. Library subroutine program card. This is the column binary equivalent of the row binary program card described in Chapter 3.

The FORTRAN Library

The entry point names, card labels, and descriptions of the subroutines that make up the library file of the Master tape are listed below. The order in which they are listed is the order of their appearance in the Editor deck immediately following 9LIB.

Control Subroutine (internal to FORTRAN)

ENTRY POINT NAME	EDITOR DECK CARD LABEL	DESCRIPTION
(FPT)	9FPT	Floating point trap routine. Handles the floating point trap feature in an object program.

Mathematical Subroutines

ENTRY POINT NAME	EDITOR DECK CARD LABEL	DESCRIPTION
EXP (1	9XP1	These are exponential subroutines compiled by a source program statement such as Y = E**X. One of the three routines is compiled, depending on whether a fixed point base-fixed point exponent, floating point base-fixed exponent, or a floating point base-floating point exponent mode, respectively, is specified in the source program statement. These subroutines are internal to FORTRAN.
EXP (2	9XP2	
EXP (3	9XP3	

ENTRY POINT NAME	EDITOR DECK CARD LABEL	DESCRIPTION
EXP	9XPF	This is a floating point exponential subroutine compiled into the object program by a source program statement such as A = EXPF(X).
LOG	9LOG	Floating point natural log subroutine.
ATAN	9ATN	Floating point arctangent subroutine.
SIN, COS	9SCN	Floating point sine and cosine subroutine.
TANH	9TNH	Floating point hyperbolic tangent subroutine.
(IFMP), (IFDP)	9CAS	Complex arithmetic basic routine. Performs complex multiplication and division.
IABS	9CAB	Complex absolute value routine. For the complex number x + iy, IABS computes the square root of $(x^2 + y^2)$.
IEXP	9CXP	Exponential subroutine for complex numbers. If z = x + iy, IEXP computes e^z.
ILOG	9CLG	Natural logarithm subroutine for complex numbers. If z = x + iy, ILOG computes $\log_e z$.
ISQRT	9CSQ	Square root subroutine for complex numbers. If z = x + iy, ISQRT computes $(z)^{1/2}$.
ISIN, ICOS	9CSC	Sine and cosine subroutine for complex numbers.
(DFAD), (DFSB), (DFMP), (DFDP)	9DPS	Double-precision basic routine. Performs double-precision addition, subtraction, multiplication, and division.
DMOD	9DMD	Double-precision modulo routine. If x and y are double-precision numbers, DMOD computes $x - [x/y]\ y$, where $[a]$ is the integral part of a.
DINT	9DIN	Obtains integral part of a double-precision number.
IXPP(2, DEXP(2	9DX2	Double-precision and complex exponential subroutine.
DEXP(3	9DX3	One of two routines is compiled depending on whether a floating point base - fixed point exponent or a floating point base - floating point exponent is specified.
DEXP	9DXP	Exponential subroutine for double-precision numbers.
DLOG, DLOG10	9DLG	Natural logarithm and common logarithm subroutines for double-precision numbers.
DSQRT	9DSQ	Square root subroutine for double-precision numbers.
DSIN, DCOS	9DSC	Sine and cosine subroutine for double-precision numbers.
DATAN, DATAN2	9DAT	Arctangent subroutine for double-precision numbers.
SQRT	9SQR	Floating point square root subroutine for single-precision, real numbers.

Input/Output Subroutines (internal to FORTRAN)

ENTRY POINT NAME	EDITOR DECK CARD LABEL	DESCRIPTION
(TSB), (RLR)	9TSB	Reads binary tape record(s) into storage.
(STB), (WLR)	9STB	Writes binary tape record(s) from storage.
(CSH)	9CSH	Reads alphameric card(s) into storage and converts their contents to BCD.
(TSH), (TSHM)	9TSH	Reads BCD tape record(s) into storage.
(STH), (STHM), (STHD)	9STH	Writes BCD tape record(s) from storage.
(SCH)	9SCH	Punches alphameric card(s) from storage.
(SPH)	9SPH	Prints information from storage on the on-line printer.
(BST)	9BST	Backspaces the designated tape one record.
(SDR), (DRS)	9DRM	Writes or reads data on the designated drum. (SDR) is the write section of 9DRM. (DRS) is the read section of 9DRM.
(EFT)	9EFT	Writes an EOF mark on the designated tape.
(RWT)	9RWT	Rewinds the designated tape.
(SLI)	9SLI	Controls input of lists containing nonsubscripted array names.
(SLO)	9SLO	Controls output of lists containing nonsubscripted array names.
(RER), (RDC)	9RER	Error routines for tape reading.
(WER), (WTC)	9WER	Error routines for tape writing.
(IOB), (EXB)	9IOB	Controls input/output of binary data.
(IOH), (FIL), (RTN)	9IOH	Controls input/output and conversion of alphameric data.
(IOS), (RDS), (WRS), (BSR), (WEF), (REW), (ETT), (RCH), (TEF), (TCO), (TRC)	9IOS	Supervisory control of channel-unit designation during input/output.
(IOU)	9IOU	DSU Channel-Unit table.

Monitor Subroutines

ENTRY POINT NAME	EDITOR DECK CARD LABEL	DESCRIPTION
CHAIN	9CHN	CHAIN locates chain links, loads the chain executive loader into lower storage, and transfers control to it.
DUMP PDUMP	9DMP	DUMP takes storage dumps according to the specifications in the arguments A, B, and C; it positions the System tape at the Sign-On record; and it restores 1-CS to begin the next job. PDUMP takes storage dumps as specified by the argument A, B, and C; it restores the condition of the machine; and it returns to the program that called it. The MQ and locations 0, 1, and 2 are saved and restored.

Other Subroutines

ENTRY POINT NAME	EDITOR DECK CARD LABEL	DESCRIPTION
XLOC	9XLO	The source statement, L = XLOCF(N), returns the relocated location of its argument (variable N) to the accumulator as a FORTRAN fixed point constant.
(EXE)	9EXE	(EXE) controls the object program error procedure when execution is not under Monitor control.
(EXEM)	9EXEM	(EXEM) controls the object program error procedure when execution is under Monitor control.
EXIT	9XIT	EXIT positions the System tape at the Sign-On record and restores 1-CS to begin the next job.
(TES)	9TES	The instruction XEC* $(TES) may be used in a FAP-coded subroutine to make sure that the execution of any previous FORTRAN WRITE statement is complete and checked. (TES) is set to TSX (WER), 4 by 9STH and 9STB and is reset to NOP by 9WER.

Each FORTRAN source program input/output statement requires one or more library subroutines during execution of the object program. In order to save both compilation time and card searching time caused by duplication of library subroutines (as might easily occur in a main program-subprogram sequence), binary card copies (from the Editor deck) of certain of these frequently used subroutines may be kept aside and inserted into the load deck when needed.

For programs loaded under Monitor control, missing library routines are automatically supplied.

The library subroutines required for each source input/output statement are as follows:

FORTRAN Source Input/Output Statement	The Editor Deck Card Labels of the Library Subroutines Required
READ	9CSH, 9IOH, 9IOS, 9IOU, 9TES, 9EXE
READ INPUT TAPE	9TSH, 9RER, 9IOH, 9IOS, 9XIT, 9TES, 9EXE, 9IOU
WRITE OUTPUT TAPE	9STH, 9WER, 9IOH, 9IOS, 9IOU, 9TES, 9EXE
PUNCH	9SCH, 9IOH, 9IOS, 9IOU, 9TES, 9EXE
PRINT	9SPH, 9IOH, 9IOS, 9IOU, 9TES, 9EXE
WRITE TAPE	9STB, 9WER, 9IOB, 9IOS, 9IOU, 9TES, 9EXE
READ TAPE	9STB, 9RER, 9IOB, 9IOS, 9IOU, 9TES, 9EXE
BACKSPACE	9BST, 9IOS, 9IOU, 9TES, 9EXE
END FILE	9EFT, 9IOS, 9IOU, 9TES, 9EXE
REWIND	9RWT, 9IOS, 9IOU, 9TES, 9EXE
WRITE DRUM	9DRM
READ DRUM	9DRM

In addition to the above listed subroutines, the following two subroutines may or may not be required, as determined by the object program storage map:

Subroutine	Entry Point Name	Label
Short List Input	(SLI)	9SLI
Short List Output	(SLO)	9SLO

The library subroutine Floating Point Trap, 9FPT, must always be included in the object program deck when loading the program for execution.

FORTRAN floating point overflow/underflow will be treated as follows:

Overflow Occurs in:	9FPT Returns:
AC	All 1's in the AC
MQ	All 1's in the MQ
AC and MQ	All 1's in the AC and the MQ

Underflow Occurs in:	9FPT Returns:
AC	All 0's in the AC
MQ	All 0's in the MQ
AC and MQ	All 0's in the AC and the MQ

NOTE: If library subroutines are not punched out at compile time, it is recommended that the section of the object program storage map labeled "Entry Points to Subroutines Not Punched from Library" be examined to insure that the library decks loaded with the object program are complete. However, if execution is under Monitor control, a library search will be performed.

Library Subroutine Program Card

The first card of every library subroutine must be a program card that has the same format as the program card described in Chapter 3.

Additional entry points of library subroutines, each of which is represented by a distinct name, may be designated as primary or secondary. If column 37 of the entry word is punched, it is a secondary name; if not, it is a primary name. The meaning of primary and secondary names arises out of the following rule of precedence which is used by FORTRAN in compiling library subroutines into the object program.

Rule: When a function is mentioned in a source program, the routine that will be used is the first routine in the System tape library that meets either of the following conditions:

1. The name mentioned is a primary name of the routine.
2. The name mentioned is a secondary name of the routine and at least one of the primary names of the routine is also mentioned.

If the System tape library is arranged with subroutines that have many secondary names preceding subroutines that have few names or none, the above stated rule will prevent unnecessary duplication of subroutines in the object program.

Example: Suppose the System tape contains an arcsine routine that has an entry point to a routine to compute a square root, and this routine is given two names, ASINF (primary) and SQRTF (secondary). Also, subsequently on the System tape there is an ordinary square root subroutine with the single name SQRTF (primary). A source program that requires both ASINF and SQRTF will cause only the former program to be included in the object program.

FORTRAN Channel-Unit Tables

The FORTRAN Processor contains two channel-unit tables: the IOU table contains tape assignments for FORTRAN object programs; the IOPU table contains tape assignments for the FORTRAN Processor.

Description of DSU Channel-Unit Table for FORTRAN

In the library section of the FORTRAN Editor deck there are two relocatable binary cards labeled 9IOU0000 and 9IOU0001. The first of these is the program card; the other card contains the DSU Channel-Unit table for FORTRAN object programs.

The table assigns tapes as follows:

FORTRAN Tape Number	Channel-Unit
1	A 1
2	B 2
3	B 3
4	A 4
5	A 2
6	A 3
7	B 4
8	B 1

From this assignment it can be seen that provision has been made for operation on two channels. Table A shows the symbolic cards which produced 9IOU0000 and 9IOU0001.

To change the tape assignment, the following procedure is necessary:

1. Change the appropriate cards in a symbolic deck for 9IOU. Table B indicates this procedure by showing the following example for a 7090 or 7094 installation with 12 tape units and 3 channels.

FORTRAN Tape Number	Channel-Unit
1-4	A 1-4
5-8	B 1-4
9-12	C 1-4

2. FAP-assemble this symbolic deck.
3. Replace the two cards labeled 9IOU0000 and 9IOU0001 in the Library portion of the Editor deck by the two cards produced from the FAP assembly.
4. Generate a new System tape.

It is recommended that the table contain zero entries for those units not actually available. Then a machine stop (i.e., HPR 0,6) will occur during execution of an object program if a unit addressed has been designated as not available on the machine. If this stop occurs, enter the instruction AXT n,4; where n is the unit/channel address desired as it would appear in the Unit table (for example, 1201), and depress the Start key. Under Monitor control, EXEM handles this error.

Entries may be made in the Unit table for tapes on any of the eight channels A-H. Likewise, the Reader, Punch, and Printer may be changed from channel A to any other channel by making the appropriate changes in the first three entries of 9IOU.

Description of IOP Channel-Unit Table

Tape assignments for the FORTRAN Processor are contained in the Input/Output Package Channel-Unit table (IOPU). Card 9FIOPU01 in the Editor deck contains the following tape assignments:

FORTRAN Tape Number	Channel-Unit
1	A 1
2	B 2
3	B 3
4	A 4
5	A 2
6	A 3
7	B 4
8	B 1
9	A 5
10	B 5
11	A 6
12	B 6
13	A 7
14	B 7
15	A 8
16	B 8

```
*       FAP
        COUNT     30
*       FORTRAN LIBRARY / INPUT-OUTPUT CHANNEL-UNIT
*       TABLE / 9IOU.
*
        ENTRY     (IOU)
        REM     . . . . . . . . . . . . . . . . . . . . . . . . .
        PZE       PRINT
        PZE       PUNCH
        PZE       READ
 (IOU)  PZE       NTAPES
        PZE       A1
        PZE       B2
        PZE       B3
        PZE       A4
        PZE       A2
        PZE       A3
        PZE       B4
        PZE       B1
* INSERTIONS AND/OR DELETIONS SHOULD BE MADE BETWEEN
* NTAPES AND (IOU).
NTAPES EQU        *-(IOU)-1
*
 READ   BOOL      1321
 PUNCH  BOOL      1341
 PRINT  BOOL      1361
 A1     BOOL      1201
 A2     BOOL      1202
 A3     BOOL      1203
 A4     BOOL      1204
 B1     BOOL      2201
 B2     BOOL      2202
 B3     BOOL      2203
 B4     BOOL      2204
        REM     . . . . . . . . . . . . . . . . . . . . . . . . .
        END
```

Table A. DSU Channel-Unit Table as Found in the 7090/7094 FORTRAN II Input-Output Library

Location OPTION should be changed to NOP to activate the setting of tape densities. Tape density table PKUP-1 currently sets density on 16 tapes: tapes 5, 6, 7 are set to low density; the others are set to high density. Room is available in the table for additional entries. A density setting may be changed by inverting the prefix sign in the entry for each tape.

```
*       FAP
        COUNT     40
*       FORTRAN LIBRARY / INPUT-OUTPUT CHANNEL-UNIT
*       TABLE / 9IOU.
*
        ENTRY     (IOU)
        REM     . . . . . . . . . . . . . . . . . . . . . . . . .
        PZE       PRINT
        PZE       PUNCH
        PZE       READ
 (IOU)  PZE       NTAPES
        PZE       A1
        PZE       A2
        PZE       A3
        PZE       A4
        PZE       B1
        PZE       B2
        PZE       B3
        PZE       B4
        PZE       C1
        PZE       C2
        PZE       C3
        PZE       C4
* INSERTIONS AND/OR DELETIONS SHOULD BE MADE BETWEEN
* NTAPES AND (IOU).
NTAPES EQU        *-(IOU)-1
*
 READ   BOOL      1321
 PUNCH  BOOL      1341
 PRINT  BOOL      1361
 A1     BOOL      1201
 A2     BOOL      1202
 A3     BOOL      1203
 A4     BOOL      1204
 B1     BOOL      2201
 B2     BOOL      2202
 B3     BOOL      2203
 B4     BOOL      2204
 C1     BOOL      3201
 C2     BOOL      3202
 C3     BOOL      3203
 C4     BOOL      3204
        REM     . . . . . . . . . . . . . . . . . . . . . . . . .
        END
```

Table B. Example of Channel-Unit Table for 7090/7094 Installation with 12 Tapes

FORTRAN ENTRY POINT NAMES	Adapted From		Number of Locations (Decimal)		Error	Description
	SHARE Routine	Distribution Number		Erasable		
ATAN	IB ATN1	507	77	+ 3	$\leq 1 \times 10^{-8}$	Computes in radians the principal value of arctan (x) where x is a floating point single-precision argument
EXP	IB FXP	507	46	+ 4	$\leq 1 \times 10^{-8}$	Computes e^x for a floating-point single-precision argument $\vert x \vert < 88.028$
SIN COS	IB SIN1	507	105	+ 4	$\leq 1 \times 10^{-8}$	Computes the sine or cosine of a single-precision normalized floating-point argument x,. in radians, where $\vert x \vert < 2^{36}$
TANH	IB TANH	507	86	+ 4	$\leq 3 \times 10^{-8}$ for $x \sim .00034$ or $\sim .17$ Otherwise $\leq 1 \times 10^{-8}$	Computes tanh (x) for any single-precision floating-point argument, in radians
LOG	IB LOG3	665	46	+ 3	$\leq 3 \times 10^{-8}$	Computes $\log_e x$, natural logarithm, for a single-precision floating-point argument in normalized from, $\vert x \vert > 0$
SQRT	IB SQ1	721	44	+ 4	$\leq 1 \times 10^{-8}$	Computes the square root of a single-precision floating-point argument $x > 0$

7090/7094 FORTRAN II Library Math Routines

The FORTRAN Diagnostic System

The FORTRAN Compiler Diagnostic may be considered as having two divisions: The Section 1 Diagnostic and the General Diagnostic.

Section 1 Diagnostic

The Section 1 Diagnostic is designed to detect as many errors as possible on the language or syntactical level. It results from the statement by statement scan of the entire program that Section 1 performs.

1. When a source program error is detected, the statement in error is written on tape A3 along with a message explaining the error. The message is also written on-line if END card setting 3 is 1. Section 1 then proceeds to its scan of the next statement. This means that only the first error in any one statement is detected. When all statements have been scanned, a stop occurs indicating that the Section 1 Diagnostic is complete. The diagnostic message is also written on the output tape following the source program. The source program must be corrected and recompiled.

When not under Monitor control, the source program file on tape B2 may not be used because the end-of-file mark on the source program will have been overwritten by the diagnostic message.

2. If only a machine error is detected, a printout occurs followed by a halt. The printout gives a procedure to follow in order to delete or rerun the compilation.

General Diagnostic

The errors covered by the General Diagnostic, its structure on the System tape, and its operation are described below:

Types of Errors: The General Diagnostic covers source program errors revealed by Sections 1' and 1'' through Section 6 of FORTRAN.

1. Section 1'' is a special diagnostic routine that is used to detect most of the source program errors not found by Section 1. These are primarily errors arising from the interrelationship of statements. Examples of such source program errors are: transfers to nonexistent statement numbers; parts of the program to which there is no path of flow; and transfers to nonexecutable statements.
2. The source program errors revealed by Sections 2 through 6 are primarily those arising from exceeding the source program size limitations. These limitations are given in IBM 7090/7094 Programming Systems: FORTRAN II, Form C28-6054-3.

General Diagnostic on the FORTRAN System Tape:

1. The General Diagnostic caller is contained in the Input/Output Package (IOP).
2. The entire fourth file of the System tape is the General Diagnostic, and contains the specific comments and relevant table lookups for the particular source program error.

Operation of the General Diagnostic: When a source error is encountered in any of the executive system records, control is transferred to the Diagnotic caller in IOP.

The Diagnostic caller writes 8,700 words of storage onto tape A3 and spaces forward to the fourth file of the System tape, the General Diagnostic.

The General Diagnostic performs a table search to determine which of its routines contains the information pertaining to the error. It then transfers to that routine.

The routine to which the transfer is made contains the message explaining the error. In addition, it may contain program instructions to obtain detailed instructions that will become part of the message. The error message is then written on tape A3. The message is also written on-line if END card setting 3 is 1. In the single compile mode, the message is written on tape B2 if Sense Switch 3 is up; if Sense Switch 3 is down or the END card setting is 1, the message is also printed on-line.

The General Diagnostic then transfers control to the Source Error Record.

CHAPTER 3: FORTRAN II COMPILER OUTPUT

When only the FORTRAN Compiler is being used (operations are not initiated by the Monitor Start card), the following output is obtained.

Composition of the Object Program Deck

The binary card output of a main program compilation consists of the following sequence of cards:

BSS loader (nine cards)
Program card
Program in relocatable binary
Transfer card (9 punch in col 1)

The binary card output of a SUBROUTINE or FUNCTION subprogram compilation consists of:

Program card
Program in relocatable binary

When a library subroutine is called for during a compilation in which the punching of subroutine cards was requested, the subroutine card output will be:

Program card
Program in relocatable binary

Preparing the Object Program Deck for Loading

In order to run the object program, the FORTRAN object program deck must consist of the following sequence:

BSS loader (nine cards)
Program card
Program in relocatable binary
Program card
Program in relocatable binary
Program card
Program in relocatable binary
.
.
.
.
.
Transfer card (9 punch in col 1)

One of the programs must be a main program and all of the others, subprograms. The programs may be in any order.

All the subprograms called for in the main program and in other subprograms must be in the deck if it is to be executed. Of course, they normally will be in the deck as a result of a compilation with Sense Switch 5 DOWN. If any subprograms are missing, a stop at 77733_8 or 77746_8 will occur during loading with the BSS loader.

Subprograms in the Object Program Deck

Although duplicate subprograms taken from the Library tape will never occur in a single program, a SUBROUTINE subprogram, or a FUNCTION subprogram compilation, they may easily occur in a main program-subprogram sequence.

If two subroutines in the object program deck have identical entry points, only the first will be loaded. However, if at least one entry point of the second subroutine differs from all the entry points of previously loaded subroutines, it also will be loaded. In the case where both subroutines are loaded and reference in the object program is made to an entry point that appears in both subroutines, the first subroutine loaded will be called. For example, if subroutine 1 has the entry points SIG, ALP, and BET, and subroutine 2 has the entry points SIG, ALP, and PHI, reference to SIG would cause subroutine 1 to be called, assuming that subroutine 1 is loaded first.

If a job contains the compilation or assembly of a subroutine with the same entry points as a subroutine that is loaded with the job, both subroutines will be loaded, but the compiled (or assembled) subroutine will be called during execution.

The physical sequence of subprograms belonging to any program, SUBROUTINE subprogram, or FUNCTION subprogram compilation is the reverse of their appearance in the section of the object program storage map labeled "Subroutines Punched from Library."

In order to save both compilation time and card searching time caused by duplications of library subroutines, binary decks of certain frequently used subroutines may be kept aside and inserted into the load deck when needed. This would enable some programs to be run with Sense Switch 5 UP, that otherwise could not be.

Transfer Card

The Transfer card must be the last card in the load deck. It is, however, compiled as the last card in the main program, and the main program may not be the last program in the deck ready for loading. In this case, two alternatives are available:

1. The Transfer card may simply be extracted and placed at the end of the complete load deck.
2. Another Transfer card (9 punch in col 1) may be placed at the end of the load deck. In this case a stop at 77733_8 or 77746_8 will occur during loading at the time the first Transfer card is encountered. Depressing the Start key enables loading to continue.

Binary Symbolic Subroutine Loader (BSS)

The Binary Symbolic Subroutine loader (BSS) is punched out by FORTRAN as the first nine cards of each main program. It is not, however, punched out if column binary cards for the main program are specified. The BSS loader is also not punched out with subprograms.

The 7090/7094 FORTRAN II System produces decks in either relocatable column binary or relocatable row binary form, depending on the setting of Sense Switches 1 and 4. In these relocatable binary decks, instructions are assigned to consecutive storage locations starting at 0, and all location references are relative to 0. When a relocatable binary deck is loaded, location references are altered according to the actual locations occupied by the program in storage.

All decks, both main program and subprogram, compiled by FORTRAN and punched out in relocatable binary form, are loaded by the BSS loader. The loader enables programs in relocatable row binary form to retain symbolic references to subprograms. As a result of this feature, a main program and each of its subprograms can be independently compiled. It is possible to compile a main program for which some or all of the subprograms have not yet been written. After a main program and each of its subprograms have been compiled, the resulting relocatable binary decks can be loaded together and executed. At execution time, the relocatable binary decks of the main program and its subprograms - starting at relocatable 0 - are stacked in the card reader in any order, headed by the nine loader cards. The program and subprograms are loaded and relocated by the BSS loader, using control information supplied with the compiled routines; they are then executed.

Control information for the relocation process is provided to the loader by program cards, one preceding the main program deck and one preceding each subprogram deck. A program card is the tenth card punched out by FORTRAN for a main program and the first card for each subprogram. The program card specifies the number of locations to be occupied by the routine; this number is used as an increment for relocating an immediately subsequent routine. The increments specified by successive program cards are cumulative. Program cards contain other information required by the loader to interpret symbolic cross-references between the main program and its subprograms and between levels of subprograms.

In addition to relocatable binary decks produced by FORTRAN, the loader can also load binary cards, both absolute and relocatable, produced by a system other than FORTRAN.

Transfer Card

The last card of the last deck to be loaded must be a Transfer card. Row 9, column 1, must be punched on the Transfer card, and row 9, columns 2-36, must be blank. The rest of the card is ignored. This card will signal the BSS loader that the main program and all subprograms have been loaded. FORTRAN produces a Transfer card as the last card of a main program deck.

Transfer List

If a program refers to subprograms, the first instruction of the program in the compiled deck is preceded by a Transfer list. This Transfer list consists of the names of all the subprograms referred to in the program; in the case of FAP subprograms with more than one entry point, a name is listed for each entry point to which reference is made. Note that subprograms as well as main programs may have a Transfer list, since a subprogram may call for lower level subprograms.

When a program has a Transfer list, the first name in the list occupies relocatable location 0 in the compiled deck, and each name in the list is counted as one word of the program.

Execution of the BSS Loader

Initially, the BSS loader loads itself into core storage; the first three words are loaded into the first three core storage locations, and the remaining words, into locations 77453_8 to 77777_8.

The first loader pass is then executed. In the first pass, absolute locations are assigned to instructions, data, and Transfer list names of the subroutines being loaded. A symbol table is set up in which each subprogram name is associated with the absolute location of the entry point designated by the name.

In the second pass, each Transfer list name is replaced by an instruction which transfers control to the entry point designated by the name. The symbol table provides the necessary information for this step. Execution of the main program then begins.

(The loader leaves parameters in a location for use by WD IOF, SHARE Distribution No. 978. This distribution is a package of buffered input/output routines for FORTRAN.)

Card Formats Acceptable to the BSS Loader

The following types of cards may be loaded by the BSS loader:

1. Program cards
2. Transfer cards (indicating end of the first loader pass)
3. Control cards
4. Absolute transfer cards
5. Absolute binary instruction and data cards
6. Relocatable binary instruction and data cards
7. Relocatable binary transfer cards
8. A self-loader that loads at least 32 words

Blank cards will be ignored. The following is a description of the formats required for the types of cards listed above.

Program Card

Row	Column	Description
9	1	Must be punched.
	2	Must be blank.
9	3	The checksum will be ignored if it is blank or if there is a punch in column 3.
	4-18	Count of words on this card, excluding the 9-row.
	19-21	Ignored.
	22-36	Must be blank.
	37-72	Checksum (add-and-carry-logical) of all words on this card except 9R.
8	1-3	Ignored.
	4-18	Contains the number of words in the Transfer list for this program. This list must be loaded next and followed by instructions in the usual relocatable format. This field will be zero when the program being loaded does not require subprograms for its execution and, therefore, has no Transfer list.
	19-21	Ignored.
	22-36	Contains a number showing the length of lower storage. This is the program break. It is the same as the address-plus-one, relative to zero, of the last word of the program, excluding data assigned to common storage. All location references in the address or decrement fields of instructions being placed in storage are relocated as either lower storage or upper storage locations, depending on the range in which they fall with respect to the program break <u>and</u> on the associated relocation digits. References to be relocated as lower locations are relocated by the current increment of the loader. The current increment is increased by the number in this program card field when the next control card or program card is read.

Row	Column	Description
8	22-36	References to be relocated as upper locations are relocated according to the information provided to the loader by the last preceding control card.
	37	Punched if the program is to be loaded starting at an even storage location (pertains to Monitor operations only; ignored by BSS Loader).
	38-57	Ignored.
	58-72	This is the first location below the common storage area required for the program. This field must be blank (zero) if no common data is assigned. In the case of machine language subprograms, common data storage may be assigned downward from 77777_8; this address is then the 2's complement of the length of common storage. In the case of 709 and 7090 FORTRAN programs, common data storage is always assigned downward from 77461_8. Unless otherwise indicated, the loader will cause the common data of successive routines to be overlapped. If overlapping is not desirable, the loader's current decrement should be reset by an entry in the 9R address of the control card placed in front of the program card for the routine whose data is to be moved down. The control card sets a new decrement into the loader to cause relocation of upper storage on cards following the program card. The decrement will be retained until replaced by a new decrement given on a subsequent control card.
7	1-36	If the program is a subprogram, this field contains the BCD representation of the name assigned to the first entry point (or to the subprogram if there is only one entry point). If the name contains fewer than six characters, each unused 6-digit group at the right must be filled in with the BCD character 110000_2. If the program is a main program, this field must be blank, as a main program is considered to have a blank name.
	37	A punch in this column indicates that this entry point is secondary. No punch indicates that this entry point is primary.
	38-57	Ignored.
7	58-72	The address, relative to zero, associated with the name in columns 1-36.
6,5,..., 0,11,12	1-36	If the program is a FAP subprogram with more than one entry point, the names assigned to the second, third, etc., points are listed in these fields in order, i.e., the second in row 6, the third in row 5, etc. When all names have been listed, the remaining rows are left blank. The names are represented as described for row 7.
	37	A punch in this column indicates that this entry point is secondary. No punch indicates that this entry point is primary.
	38-57	Ignored.
	58-72	The address, relative to zero, associated with the name in columns 1-36.

NOTE: If there are more than ten entry point names, one or more additional program cards are required, containing the eleventh, twelfth, etc., names. Supplementary program cards must have row 9 punched as specified in the description of the program card, and the names must continue in row 8.

Transfer Card

Row	Column	Description
9	1	Must be punched.
	2	Must be blank.
	3	Ignored.
	4-36	Must be blank.
	37-72	Ignored.
8,7,..., 0,11,12	all	Ignored.

Control Card

Row	Column	Description
9	1	Must be blank.
	2	Must be punched.
	3	Ignored.
	4-12	Must be blank.
	13	Must be punched.
	14-18	Must be blank.
	19-21	Ignored.
	22-36	Contains the number of locations to be added to the current increment of the loader to yield a new increment. The new increment is effective for the relocation of lower storage locations in the next routine loaded.
	37-57	Ignored.
	58-72	Contains the 2's complement of the number the loader is to use in relocating common data downward. This number becomes the current decrement of the loader. The current decrement is reset each time a control card is read and is not related to the last previous decrement of the loader.
8,7,..., 0,11,12	all	Ignored.

NOTE: When a control card is read, the increment of the loader is increased by the number given in row 9, columns 22-36, of the last preceding program card, and further increased by the number in row 9, columns 22-36, of the control card. The decrement of the loader, however, is reset to a value not related to the previous decrement. If row 9, columns

58-72, of the control card is blank, the decrement will be reset to 0. It is for this reason that common storage for different programs overlap.

Absolute Transfer Card

Row	Column	Description
9	1-21	Must be blank, except that a punch in column 3 will be ignored.
	22-36	Absolute location to which the BSS loader is to transfer control.
	37-72	Ignored.
8,7,..., 0,11,12	all	Ignored.

Absolute Binary Instruction or Data Card

Row	Column	Description
9	1-2	Must be blank.
	3	If punched, the checksum will be ignored.
	4-13	Must be blank.
	14-18	Count of words on this card, excluding row 9.
	19-21	Ignored.
	22-36	Address into which the first (i.e., 8L) word is to be loaded.
	37-72	Checksum (add-and-carry-logical) of all words on this card except 9R.
8,7,..., 0,11,12	Instructions or data to be loaded.	

Relocatable Binary Instruction or Data Card

Row	Column	Description
9	1	Must be blank.
	2	Must be punched.
	3	If punched, the checksum will be ignored.
9	4-18	Count of words on this card, not including rows 8 and 9.
	19-21	Ignored.
	22-36	Address, relative to zero, into which the first (i.e., 7L) word is to be loaded.
	37-72	Checksum (add-and-carry-logical) of all words on this card except 9R.
8	1-72	Both words are read together and contain information about the relocation of location reference in the address fields (columns 22-36 or 58-72) or decrement fields (columns 4-18 or 40-54) of the instructions in rows 7-12 of this card.
7,6,..., 0,11,12	Instructions or data to be loaded.	

The digits in row 8 of this card are interpreted one at a time and are related to the decrement field of 7L, the address field of 7L, the decrement field of 7R, the address field of 7R, the decrement field of 6L, etc. The binary digits have the following significances:

Row	Column	Description
	0 -	Ignore this field.
	10 -	If the number in this field is equal to or greater than the program break, relocate as an upper storage location. If the number is less than the program break, relocate as a lower location.
	11 -	If the number in this field is equal to or greater than the program break, relocate as a lower storage location. If the number is less than the program break, relocate as an upper location.

Instructions for which there is no room in the 8-row for the necessary relocation digits must be put on another card.

Relocatable Binary Transfer Card

Row	Column	Description
9	1	Must be blank.
	2	Must be punched.
	3	Ignored.
	4-21	Must be blank.
	22-36	Relocatable entrance address (in octal).
	37-72	Must be blank.
8,7,6,..., 0,11,12	all	Must be blank.

A Self-Loader which Loads at Least 32 Words

The 9-left decrement of the self-loader must be greater than 32_{10}.

CHAPTER 4: FORTRAN II MONITOR

The 7090/7094 FORTRAN II Monitor permits the following operations:

1. FORTRAN compiling.
2. FAP (FORTRAN Assembly Program) assembling.
3. Execution of object programs.
4. Execution of programs in links, a procedure necessary where the total program is too large to fit into storage and a link is a section of it which does fit into storage.

Input to the 7090/7094 FORTRAN II Monitor System consists not only of the source program, but may include the following as well:

1. FAP symbolic cards.
2. Object program cards.
3. Data cards.
4. FORTRAN Monitor control cards.

The relative order of a series of different types of input does not matter, provided that each separate deck, whether source program, object program, etc., is preceded by appropriate control cards.

The 7090/7094 FORTRAN II Compiler may be considered a subsection of the Monitor. Under FORTRAN control a single source program may be compiled. Nothing further, including execution, can be done. If multiple compilation of a series of FORTRAN source programs is desired, Monitor control is required.

Definition of Job

A job is the basic unit being processed by the Monitor at any one time; it consists of one or more programs and is either an Execute job or a Non-Execute job. As an Execute job, it is to be executed immediately after whatever processing is required. This means that the programs of the job are related to each other. A Non-Execute job contains programs which need not be dependent. Each program is processed as the control cards for the job specify. The "processing" that is given a program is one of the following:

Execute	Non-Execute
1. FORTRAN Compilation.	1. FORTRAN Compilation
2. FAP Assembly.	2. FAP Assembly (object program input is ignored).
3. Relocation of object program input.	
4. For jobs divided into links, treatments of chain links.	

A job is one of the following five types (the first three are Non-Execute jobs and the last two are Execute jobs):

1. One or more FORTRAN source programs to be compiled. This is simply multiple compilation. The programs may be main programs or subprograms.
2. One or more FAP symbolic programs to be assembled. The programs may be main programs or subprograms.
3. An intermixture of job types 1 and 2. This results in multiple compilation and assembly of FORTRAN and FAP source programs, with object program output for each source program input. There may be any combination of main programs and subprograms.
4. A sequence of input programs for immediate execution. The input programs may be of job types 1 and 2, together with relocatable column binary object program cards. Data cards, to be used during execution, follow the input programs. Input programs each consist of a single main program-subprogram sequence not larger than the available core storage. This sequence constitutes a "machine load."
5. A sequence of input programs meant for execution where each input program is a job of type 4. The data cards are placed at the end of all the input programs. This is called a Chain job and each of the jobs of type 4 is a Chain link. This permits a single object program execution to consist of more than one "machine load."

Input

Job Input Decks

The 7090/7094 FORTRAN II Monitor will process the following types of job input:

1. Multiple Compilation/Assembling. Input will consist of an indefinite number of FORTRAN or FAP source decks for compilation or assembly only. The input for each such job will contain, in the order listed:

DATE card, if any
I. D. card
Control cards for deck 1, if any
Source deck 1
Control cards for deck 2, if any
Source deck 2
.
.
.
Control cards for deck n, if any
Source deck n
End-of-File card if input is from on-line cards.

2. Compilation/Assembling and Execution. Input will consist of one main program and all its subprograms to be compiled and executed, together with the data. These programs may be a mixture of FORTRAN or FAP source programs and machine-language object programs. The object programs may be relocatable column binary cards that are the output from a previous FORTRAN compilation (with the Transfer card, if any, removed) or from a FAP assembly, or are hand-coded subroutines for FORTRAN. The input deck for a job of this type will contain, in the order listed:

DATE card, if any
I. D. card
XEQ card
Control cards for deck 1, if any
Deck 1: Object program or Source program
Control cards for deck 2, if any
Deck 2: Object or Source program
Control cards for deck n, if any

Source programs precede object programs.

No control cards for object program decks.

.
.
.

Deck n: Object or Source program
DATA card, if data follows
Data deck, if any
End-of-File card, if input is from cards on-line

3. Chain Job. Input to a Chain job consists of the Chain Link decks to be compiled and executed, together with their data decks. A Chain Link deck contains a main program together with all its subprograms, where any or all of these may be source or object decks. The Chain Link deck, then, in the order listed, contains:

CHAIN (R, T) for this particular link
Control cards for deck 1, if any
Deck 1: Object or Source program
Control cards for deck 2, if any
Deck 2: Object or Source program
.
.
.
Control cards for deck n, if any
Deck n: Object or Source program

Source decks precede object decks.

The input deck to a Chain job, in the order listed, contains:

DATE card, if any
I. D. card
XEQ card
Link deck 1
Link deck 2
.
.
.
Link deck n
DATA card
Data decks, if any
End-of-File card, if input is from cards on-line.

There is no restriction on the ordering of the Chain Link decks in the Chain Job deck, but it is most economical of execution time if links are stacked in the order in which they are called.

Each binary object program must contain at least one card other than the program card.

Preparing Jobs on Input Tape Off-Line

1. Be sure that card-to-tape equipment handles column binary cards unless no jobs include binary input.
2. Write each job on tape as a separate file.
3. In a separate file, place an End Tape card after the last card of the last job.

Preparing Jobs for On-Line Input

1. Place an End-of-File card (7- and 8-punch in column 1) after each job.
2. Place an End Tape card after the last End-of-File card.

The only difference between decks to be read on-line and those to be transcribed onto the input tape off-line is that the on-line decks must have an End-of-File card after each job.

Monitor Features

The third record of the Monitor is the "Sign-On" record. This may be programmed by the installation to handle accounting or other identifying information pertaining to a job. It reads and interprets the DATE card and the I. D. card which are the first cards for any given job. In addition, it recognizes the END TAPE card which signals that no more jobs follow. The IBM version of the Sign-On record prints the I. D. card on-line, writes it on tape for off-line printing, and signals the beginning of a job. It also prints and writes on tape the total number of lines of output at the completion of a job. This number includes output both from compilation and from execution of the job. If an installation elects to program this record, it will be useful to have certain locations left undisturbed at all times in which to save desired information. For this purpose, the Monitor leaves available locations $3\text{-}7_8$ and $11_8\text{-}137_8$.

There is a complete set of control cards for the Monitor. These are distinguished by an asterisk (*) in column 1. In general, they are of two types; one type governs the job as a whole, telling what it consists of, and the other governs output options. In addition to this set of control cards, there are the DUMP card, the START card, and the RESTART cards which are self-loading binary cards. Each of these three card types permits processing to be restarted when an unexpected stop occurs.

The FORTRAN Monitor System uses eight tapes on two channels. These are A1, A2, A3, A4, B1, B2, B3, B4. A2 is the input tape and A3 is the output tape. It should be noted that the correspondence between logical tape designations used in FORTRAN source program input/output statements and the actual tape assignments at object time is set in the Unit table (IOU) in the FORTRAN Library.

Tape B1 is used only for a Chain job or FAP updating.

Monitor control card information is written and printed on-line.

Object programs are loaded by the BSS control of the Monitor.

Monitor Control Cards

All Monitor control cards must have an "*" in column 1. With the exception of the I.D. card, the specific control instruction of the card is punched in columns 7-72. Punching may be done according to normal FORTRAN rules, which means that blanks are ignored.

Governing the Job as a Whole; Type 1 Control Cards

1. I.D. Card. This card must be present for every job, and if there is no DATE card, it must be the first card for the job. If there is a DATE card, it is first and the I.D. card immediately succeeds it. Columns 2-72 may contain anything that the installation's Sign-On record is prepared to process.
2. XEQ. This card must follow the I.D. card of a job which is to be executed.
3. DATA. This card must immediately precede the data for jobs that are to be executed. It is not needed for jobs which do not require data.
4. CHAIN (R, T). This card is used to separate links within a single Chain job and to specify the tape on which the link object program is to be stored for execution. It must precede the physically first program (or subprogram) of each Chain link, regardless of whether the program is a source or object program. R is a fixed point number greater than 0 but less than 32,768 which denotes an identifying label for the tape record which contains the link, and T is the actual unit designation of the tape on which the link is to be stored at execution time.
5. DATE. This card permits the programmer to obtain the date as an additional part of the heading of each printed page. The DATE card may appear in either or both of the following places:
   a. Preceding the I.D. card for a job. The DATE card is the only card which may precede the I.D. card.
   b. Following the Monitor START card read on-line. The date specified in this manner will be used throughout the Monitor run. However, a DATE card appearing with a job as in 5a takes precedence over the DATE card read on-line for that job only.

Following are examples of the date field, which is specified after the DATE word of the control card: 4/2/62; 6/18/61; 9/2/61. There must be two slashes (/) in the date field in addition to two characters for the year.

6. IOP. This card prevents the zeroing out of the FORTRAN Common Input/Output Package (IOP) by the FORTRAN Monitor just prior to execution, thus making it available to object programs. COMMON storage is relocated downwards to prevent overlap with IOP. For a description of the use of IOP, see the manual IBM 7090/7094 Programming Systems: FORTRAN II Input/Output Package, Form J28-6190.

Governing Compilation of Individual Programs; Type 2 Control Cards

Under Monitor control, there are two ways by which the programmer may specify his output options for FORTRAN compilations. These are the END card and the type 2 Monitor control cards. If specifications are given by both means, the Monitor control cards take precedence. In fact, the END statement which appears in the source program listing will be that fabricated by the Monitor from the control cards. For FAP assemblies, only the first two of these control cards apply. Another result of the precedence of type 2 control cards over the END card is that the END statement for programs to be compiled by the Monitor need not have options specified following the word END. It may be only the word END without specifiying options. If no specifications are given in the END statement or in Monitor control cards for a FORTRAN compilation, a standard output is produced. This consists of the following:

1. The output tape, A3, contains the information of which the first two files of a compilation in the single compile mode are composed; that is, the source program and the map of object program storage.
2. The object program in relocatable binary is stacked on tape B4 for peripheral punching without the required library subroutines.
3. The format for the B4 output tape is: one file containing the I.D. card for the job, one file containing primary program output for the job or for each link if a Chain job, and a BCD END TAPE file following the last job on the tape.

The Type 2 Monitor control cards and their effects are:

1. CARDS ROW. This card causes the Monitor to punch on-line standard FORTRAN relocatable row binary cards, preceded by a BSS loader if it is a FORTRAN main program.
2. CARDS COLUMN. This card causes the Monitor to punch on-line column binary relocatable cards (no loader).

Note that CARDS ROW and CARDS COLUMN cannot be used with the same source program.

3. LIST OR LIST8. Each of these cards causes the Monitor to write the object program in FAP-type language following the storage map. Both appear on the output tape. The LIST card produces listings in three columns without octal instruction representation; the LIST8 card produces listings in two columns with octal representation of each instruction and its relocation bits. If both cards are used, the LIST8 card takes precedence.

grammer in this way: The contents of columns 2-7 inclusive of a card are taken as the label if (a) it is the first card of the program which does not have an asterisk (*) in column 1, (b) the card has a C punch in column 1, and (c) if at least one of the columns 2-7 does not contain a blank. This label, with blanks treated as zero, is then placed in columns 73-78 of the output cards, with columns 79 and 80 used for serialization. Serialization begins with 00 and recycles when 99 is reached. If, however, the label does not use all of columns 73-78, serialization begins with zero and increases to 99...9, filling all remaining columns, through column 80, before it recycles. The Symbol table and all subroutines obtained with the program are serialized with their own names in columns 73-78.

If conditions (a), (b), and (c) do not all hold, the labeling is applied in the following way with the LABEL control card present: For a subprogram, the name of the subprogram is used; for a main program, 000000 is used in columns 73-78. The LABEL card option corresponds to END card setting 7.

Labeling may be obtained on the off-line output cards of a FAP assembly. The information in card columns 2-7 of the FAP page title card will appear as the label. Serialization will occur as above.

6. SYMBOL TABLE. This card causes the Monitor to punch a symbol table.

7. ROW. This card causes the Processor to stack relocatable row binary cards on tape B4 for peripheral punching. This option may not be exercised in an EXECUTE job. (The Monitor will delete execution if a ROW card appears in an EXECUTE job.)

8. PACK. This card causes the Processor to pack records on the off-line listing tape. There will be five 120-character lines per record.

9. DEBUG. This card follows the last source program of a job, if any, and precedes the debug cards for each job or link of a Chain job.

10. PRINT. This card causes the Processor to print on-line all information being written on the off-line listing tape.

Other Control Cards

There are three other Monitor control cards: FAP, END TAPE, and PAUSE.

1. FAP. This card is placed immediately before the FAP program cards that are input to the Monitor. It specifies that those cards are to be FAP assembled. The FAP card follows any type 2 Monitor control cards that may be used.
2. END TAPE. This card designates the end of the last Monitor job. It must be a separate file on the input tape.
3. PAUSE. This card is placed in the job input deck at any point(s) at which the programmer wishes the machine to halt during the reading of the input tape. In this way, a pause for such purposes as tape reel mounting may be obtained. Processing may be restarted by depressing the Start key.

Other cards, not strictly control cards, may be input to the Monitor.

1. Cards with an asterisk in column 1 may be included with the control cards but their information field will be treated in the manner of comments. When read, they will be printed on-line and written on tape for off-line printing.
2. End of File. This is not a Monitor control card. It is used only when input to the Monitor is on-line. When input is on-line, this card is necessary to signal the FORTRAN card-to-tape simulator to write an end-of-file mark which must separate jobs on the input tape. An End-of-File card is specified by a 7- and 8-punch in column 1. All other columns are ignored.

General Information for Monitor Operations

1. Composition of Monitor File. There are nine records in the Monitor file. In addition, two records at the end of the FORTRAN Compiler file (file 2) perform Monitor functions.

Record No.	Record Name and Description
1	Card-to-Tape Simulator. If input is on-line, this record transcribes the entire input deck onto the input tape, A2.
2	Dump. This record is called by the CALL PDUMP and CALL DUMP statements and the DUMP card.
3	Sign-On. This record handles a job at the beginning and at the end. It reads and interprets the I.D. card and the DATE card. An installation may insert its own Sign-On record coding for accounting and other tasks.
4	FAP Pass 1.
5	FAP Pass 2.
6	Monitor Scan. This record recognizes a particular type of job and transfers for proper processing. This includes processing for: a. Chain job b. Execute job c. FAP assembly d. FORTRAN compilation e. Column binary input. Monitor control cards are recognized and End cards are produced from them. When a job involves FORTRAN compilations, the

Record No.	Record Name and Description
	FORTRAN source program is prepared on tape B2.
7	Debug. This record reads the source language debugging cards and prepares debugging tables. After a program requiring debugging has been loaded by BSS Control, the program break is extended to include counter tests, calling sequences, and formats for dumping; the STR instructions are inserted in the program.
8	BSS Control 1. This makes both loader passes for all binary object programs. In addition, it sorts Chain links on respective link tapes and prepares all programs for execution. BSS Control will also search the library for any missing subroutines.
9	Library Search. This record searches the library for any missing subroutines.
10	Machine Error.
11	Source Error.
32, 33, 34	These are the same as records 7, 8, and 9.

2. When under Monitor control, a FORTRAN compilation, if desired, will produce row binary cards; however, the only cards acceptable for Monitor execution are column binary. All non-Monitor hand-coded subprograms to be used must have correct associated program cards in proper column binary form.

3. If an error occurs during any of the nonexecution phases of the Monitor, the Monitor will continue to process as much as possible of the remainder of the current job, but it will not permit execution.

a. If the error is in the source program (whether FORTRAN or FAP), an on-line printout occurs. This particular program of the job will be skipped and the next program of the job will be brought in via the Source Program Error record. WARNING: Where a nonexecution phase error occurs in any program of a job, there is the danger that succeeding programs of the job will be compiled needlessly. If the job is an XEQ job and if object programs of succeeding compiled/assembled programs are not called for by the control cards, there is no purpose in continuing to these programs. Therefore, the operator, in this case, at the time of the source program error diagnostic, should be prepared to continue to the next job by means of the appropriate RESTART card.

b. If the stop is a machine error stop, the ordinary diagnostic option will be presented by the Machine Error record. The option of continuing will enable the next program of the job to be brought in. If the job is an XEQ job, the warning given above applies here also.

c. For the case of stops not accompanied by a diagnostic message, RESTART cards and a DUMP card, which are loaded at the point of stop, are provided.

d. For unexpected stops occurring during object program execution, the DUMP or RESTART cards may be used.

4. FORTRAN Compiler diagnostics are written off-line and are printed on-line if END card setting 3 is 1. The Monitor includes diagnostics that are independent of the FORTRAN diagnostic system. On-line operator options for machine error halts, however, are given in the same way.

5. For object programs which are too large to fit into core storage, the Monitor permits the program to be executed as a sequence of smaller programs called links. The entire job is called a Chain job. Subdivision of the program into links must be done by the programmer. Instructions for programming the Chain job are given in the IBM 7090/7094 Programming Systems: FORTRAN II, Form C28-6054-3.

6. Object programs under the control of the FORTRAN II Monitor are loaded into core storage beginning at symbolic location BOTTOM (see Appendix C).

CHAPTER 5: FORTRAN II MASTER TAPE DUPLICATING PROGRAM - 9FTCP - OPERATION

Operating Instructions

The 9FTCP program is executed as follows:

1. Ready the FORTRAN II Master tape on B5.
2. Ready the copy tape on A1.
3. Ready 9FTCP in the card reader and depress the Load Cards key.
4. Upon completion 9FTCP rewinds both A1 and B5. The duplicated tape should then be file-protected.

Programmed Halts

Location (octal)	Reason for Halt	Procedure
115	Checksum for tape A1 does not agree with checksum from B5.	Depress the Start key to reread B5 and to rewrite A1.
120	RTT while reading B5.	Depress the Start key to reread B5.
122	RTT while writing A1.	Depress the Start key to rewrite A1.
125	RTT while reading A1.	Depress the Start key to reread A1.
131	B5 word count of records being copied fails to agree with that of A1.	Depress the Start key to reread B5 and to rewrite A1.
136	Final stop.	Master tape duplication completed.

CHAPTER 6: SYSTEM TAPE GENERATION UNDER MONITOR CONTROL

The FORTRAN subroutine 9EDIT and the calling program 9CLEDT, which are used to generate an up-to-date System tape, are self-contained in the Editor deck.

Operating Instructions

The following steps are necessary to generate a System tape.

Tape Assignment:

1. Ready the Master tape on B6.
2. Ready the copy tape, which will become the new FORTRAN Monitor System tape, on A6.
3. Ready the Edit tape (Editor deck on tape) on A2.
4. Ready the FORTRAN Monitor System tape on A1.

Console Operations:

1. Place the SHARE Printer Board No. 2 in the on-line printer.
2. Ready the Monitor START card in the card reader.
3. Depress the Load Cards key.

Errors During Editing:

All errors are printed on-line with recovery procedures. If a card in the Editor deck is in error, the card identification is printed on-line; however, if there is no card identification, the card origin is printed on-line.

Creating B5 Library Tape

If it is desired to create a B5 Library tape, card 9CLEDT B5 must be inserted in the Editor deck following the highest numbered card in 9CLEDT.

If, in addition, it is desired to eliminate the library on the System tape, A1, insert card 9CLEDT A1 in the Editor deck following card 9CLEDT B5.

Using Library Tape on B5

Since Section Six and BSS Control Library Search both search the Library tape, the following change must be made in the Input/Output Package (IOP):

The decrement of symbolic location (LIBT) in IOP contains the logical tape designation for the Library tape. For example, to search the library on logical tape 10 (tape B5 as distributed) change the decrement of (LIBT) to 12_8. This will cause BSS Control Library Search and Section Six to search the library on B5. However, the logical tape designation for the Library tape must be 1 or greater than 8; i.e., the library may be on A1, A5, B5, A6, or B6.

CHAPTER 7: SYSTEM TAPE GENERATION NOT UNDER MONITOR CONTROL

As a secondary mode of editing, an up-to-date System tape may be prepared on-line (not under Monitor control).

Operating Instructions

Preparation of the Editor Deck:

1. Remove all cards up to and including * DATA from the Editor deck as distributed.
2. Precede the remaining Editor deck with the row binary deck labeled EDITON, or prepare an Edit tape off-line with the remaining Editor deck.

Tape Assignment:

1. Ready the Master tape on any tape unit.
2. Ready the copy tape, which will become the System tape, on any tape unit.
3. Mount an intermediate tape on any tape unit, or if the Editor deck is on tape, mount Edit tape on any tape unit.

Console Operations:

1. Ready the Editor deck in the on-line card reader, or if the Editor deck is on tape, ready EDITON in the card reader.
2. Enter information in Console keys, as follows:

S-11	Address of copy tape (octal)
12-23	Address of Master tape (octal)
24-35	Address of intermediate or Edit tape (octal)

3. Depress the Load Cards key.

Errors During Editing:

All errors are printed on-line with recovery procedures.

Creating B5 Library Tape

If it is desired to create a B5 Library tape, card EDITON B5 must be inserted in the Editor deck following the highest numbered card in EDITON.

If, in addition, it is desired to eliminate the library on the System tape, A1, insert card EDITON A1 in the Editor deck following card EDITON B5.

Using Library Tape on B5

The procedure here is the same as it is for System tape generation under Monitor control discussed in Chapter 6.

CHAPTER 8: FORTRAN II MONITOR OPERATION

Input

Any series of job input decks as described previously.

Tape Assignment

The following tape unit assignments are to be made:

1. Ready the FORTRAN Monitor System tape on A1.
2. Ready the input tape on A2. A2 is readied regardless of whether input is on-line or off-line. The tape is not rewound by the Monitor.
3. Ready tape A3 as the output tape for listing. A3 is not rewound by the Monitor.
4. Ready A4 as a FORTRAN intermediate tape.
5. Ready B1 as a Monitor intermediate tape if executing a Chain job.
6. Ready B2 as an intermediate tape for FORTRAN compilation, PDUMP, and DUMP.
7. Ready B3 as an intermediate tape for FORTRAN compilation.
8. Ready B4 as the stacked binary output tape for off-line punching. This tape is not rewound by the Monitor.

Console Operations

The following console operations must be performed for proper FORTRAN Monitor operation.

1. Place the SHARE Printer Board No. 2 in the on-line printer.
2. For card input, ready the START card followed by the input deck(s) in the card reader.
3. For tape input, ready the START card in the card reader.
4. Depress the Load Cards key.

End of Monitor Operations

When the Monitor reaches the End Tape file on the input tape, it prints "END TAPE" on-line, off-line and as the last file on B4. The Monitor then executes a load sequence for the card reader. If the card reader is ready, but empty, an End of File is written on output tape A3. Otherwise, no End of File is written.

Tape Output

Tape A3. Each compilation has as output on tape A3:

1. The source program.
2. A map of object program storage.
3. The program in symbolic machine language if either LIST control card appeared with the source deck or the END card option $I_2 = 1$.
4. Tape statistics are written on-line and off-line, on tape A3, before BSS writes "Execution" and transfers control to the object program. Tape statistics are written for each tape on which there has been activity.

To delete tape statistics, the following preassembly patches may be made:

Sign-On -- Change the contents of HTPSTS+2 to TRA FCTWO to delete tape statistics entirely.

Change the operation fields of GTCOM+1 and STAT16+1 to NOP to delete tape statistics on-line.

BBS Control -- Change the contents of TOPR1 as follows:

TOPR1	TRA	TOPR2	to delete tape statistics entirely
TOPR1	STL	NOSTON	to delete tape statistics on-line
TOPR1	STL	NOSTOF	to delete tape statistics off-line

Each FAP assembly causes the writing of the standard assembly output on A3.

Tape B4 contains the binary output for every job on the input tape in the order in which the jobs appeared on the input tape. The binary output for each job occupies a separate file preceded by a file containing the I. D. card for the job. Each link of a Chain job occupies a separate file.

Each main program or subprogram (FORTRAN or FAP) consists of:

1. A blank card with label, if both LABEL and LIBE are requested.
2. A library, if requested.
3. A symbol table, if requested.
4. A program card.
5. The program in relocatable column binary.

NOTE: The BSS loader is not written on B4. Object programs punched from B4 may only be executed under Monitor control or with a relocatable column binary loader.

Punched Card Output (On-Line)

Object program cards for either FORTRAN or FAP source programs will be punched on-line if the CARDS ROW or CARDS COLUMN control cards accompanied the source decks.

1. FORTRAN or FAP main program compilation.
   a. Row binary output
      - BSS loader
      - Library, if requested
      - Symbol table, if requested

      } FORTRAN only
      - Program card
      - Program in relocatable row binary
      - Transfer card

   b. Column binary output
      - Library, if requested
      - Symbol table, if requested

      } FORTRAN only
      - Program card
      - Program in relocatable column binary
2. FORTRAN or FAP subprogram in row or column binary
   - Library, if requested
   - Symbol table, if requested

   } FORTRAN only
   - Program card
   - Program in relocatable binary

CHAPTER 9: FORTRAN COMPILER OPERATION (SINGLE COMPILE ONLY)

The following applies when operations are not initiated by the Monitor START card.

Input

The input to the FORTRAN Compiler may be either source statement cards or a tape prepared from the source statement cards on off-line card-to-tape equipment.

Tape Assignment

The following tape unit assignments are to be made:
1. Ready the FORTRAN Monitor System tape on A1.
2. Ready the input tape on B2. B2 is readied regardless of whether input is on-line or off-line. This tape will contain FORTRAN output for listings.
3. Ready tapes A3, A4, and B3.

Console Operations

The following operations must be performed for proper FORTRAN compiling:
1. Place the SHARE Printer Board No. 2 in the on-line printer.
2. For card input, ready the cards in the card reader.
3. For tape input, the card reader must also be readied.
4. Depress the Load Tape key to begin compiling.

At the end of a successful FORTRAN compilation:
1. The on-line or off-line output that has been called for will be completed.
2. A load-key sequence for the card reader will be executed. The card reader will be selected.
3. If no other cards are ready, the operator must depress the Start key.
4. An end-of-compilation message is printed on-line.

Failure to complete compilation will be indicated by an error comment on the on-line printer.

Sense Switch Settings

Sense switches should be set as follows for FORTRAN II compilation:

Sense Switch 1	UP	Binary cards for the object program are punched on-line. Tape B3 also contains the binary program. Tape B4 will contain no binary programs.
	DOWN	Binary cards for the output program are not punched. Tape B3 contains the binary program for the source program compiled.
Sense Switch 2	UP	Two files are produced on tape B2 for the source program compiled; they consist of the source program and a map of object program storage.
	DOWN	For each program compiled (see above), the object program in symbolic machine language is added to tape B2.
Sense Switch 3	UP	No on-line listings are produced.
	DOWN	Lists on-line the first two files of tape B2.
Sense Switch 4	UP	No column binary cards for the object program are punched. Relocatable row binary cards are punched if Sense Switch 1 is UP.
	DOWN	Column binary cards for the object program are punched if Sense Switch 1 is UP.
Sense Switch 5	UP	Library routines are not punched out or written on tape B3.
	DOWN	Library routines are punched on-line or written on tape B3, depending on whether Sense Switch 1 is in the UP or DOWN position.
Sense Switch 6		Not used.

Tape Output

FORTRAN produces the following tape output (if Sense Switch 1 is DOWN);

1. B2 (for off-line printing) contains:
   File 1 Source program
   File 2 Map of object program storage
   Program in symbolic machine language (if Sense Switch 2 is DOWN)
   NOTE: The printer must be on Single or Double space for off-line printing.
2. B3 contains binary output as follows:

   Main Program consisting of:

   Program card
   Program in relocatable binary
   Transfer card in relocatable binary
   Or
   Subprogram consisting of:

   Program card
   Program in relocatable binary

NOTE: The BSS loader is not written on B3. Each B3 record is a FORTRAN relocatable row binary card image with the exception that the 9-left row of the image contains the column binary bits (bits 10 and 12 of the 9-left row) to enable off-line punching of the cards. It should be noted that these bits have been added to the check sums of the card images.

Punched Card Output

If Sense Switches 1 and 4 are UP, FORTRAN produces the following punched card output:

1. Compilation of a main program:
   BSS loader (nine cards)
   Program card
   Program in relocatable binary
   Transfer card (9 punch in col 1)
2. Compilation of a subprogram:
   Program card
   Program in relocatable binary

NOTE: The BSS loader and the Transfer card are not punched out for subprograms.

CHAPTER 10: ERROR HALTS IN THE BSS LOADER

Location (octal)	Reason for Halt	Procedure
77432	Instructions and data overlap.	Terminate loading. Combination of instructions and common data too long. Rewrite program.
77613	Checksum error on cards.	Depress the Start key to accept information.
77615	I/O check indicator ON.	Start entire loading process over.
77616	EOF at the card reader.	Depress the Start key to read cards.
77672	Instructions overlap the symbol table of the Loader.	Terminate loading. Combination of program and transfer vectors too long. Rewrite program.
77733	More than 20 subroutines are missing.	If missing subroutines are immediately available, ready them in the card reader and depress the Start key to read the cards. See items (a) and (b) below for further description of this stop.
77746	Missing subroutines.	This stop indicates that the Transfer card has been reached. It is caused by one of two occurrences:

(a) Loading has been completed, but at least one of the subroutines called for is missing. Location 77432_8 contains the BCD name of the first missing subroutine, location 77433_8, the second, etc. The list of missing subroutine names is terminated by a word of zeros. If the missing subroutines are immediately available, they may be loaded without starting the entire loading process over again. Place another Transfer card (9 punch in col 1) at the end of the routines, ready them in the card reader, and depress the Start key.

(b) The Transfer card was encountered prematurely and should have been withdrawn. Be certain that a Transfer card is the last card at the end of the deck and depress the Start key.

CHAPTER 11: OBJECT PROGRAM HALTS

There are 16 standard error stops in FORTRAN object level input/output routines. Under Monitor control, EXEM handles these error stops by providing error messages and, where appropriate, operator options. When execution is not under Monitor control, the error stops can be identified by the HPR instruction in the storage register.

Halt	Reason for Halt	Procedure
HPR 0, 1	Illegal control character in FORMAT statement.	Depress the Start key to treat as end of format.
HPR 1, 1 HPR 2, 1 HPR 3, 1 HPR 4, 1	Illegal character in data field.	Depress the Start key to cause that character to be treated as a zero.
HPR 5, 1	Illegal character encountered in octal input data.	Depress the Start key to treat as zero.
HPR 0, 2	Illegal data card character.	Correct card. Ready it in the card reader and depress the Start key.
HPR 1, 2	EOF at card reader.	Ready more cards and depress the Start key to continue reading.

Halt	Reason for Halt	Procedure
HPR 0, 3	I/O check light turned on by last read instruction.	Depress the Start key to continue.
HPR 1, 3	Tenth consecutive failure in reading tape.	Depress the Start key to read once more.
HPR 1, 4	Physical record size exceeds buffer size.	Depress the Start key to ignore the remainder of the words in the record.
HPR 2, 4	List exceeds logical record length.	Depress the Start key to store zeros in the remaining list items.
HPR 2, 3	EOF in reading tape.	Depress the Start key to read the next file.
HPR 0, 5	I/O check light turned on by last write instruction.	Depress the Start key to continue.
HPR 1, 5	Fifth consecutive failure in writing tape after erasing.	Depress the Start key to erase and write once more.
HPR 2, 5	Failure in erasing tape (bad spot).	Depress the Start key to erase once more.
HPR 3, 5	End of Tape encountered while writing tape.	MQ contains the tape address. Dial the new tape and depress the Start key to continue writing.
HPR 0, 6	Unit address not found in Unit table (IOU).	To proceed, enter the instruction AXT n, 4; where n is the unit address as it would appear in the Unit (IOU) table (for example: 1201).
HPR 1, 7	Exponent exceeds range of DEXP.	Depress the Start key to treat e^x as x.
HPR 2, 7	Negative argument illegal for DSQRT.	Depress the Start key to treat negative argument as positive argument.
HPR 3, 7	Zero or negative argument illegal for DLOG.	For x = 0, depress the Start key to treat log x = 0. For negative argument, depress the Start key to treat as positive argument.

NOTE: The input/output routine, in which any of the above stops are encountered, can be recognized by the tag of the HPR, as follows:

Tag	Routine Labeled
1	9IOH
2	9CSH
3	9RER
4	9TSB
5	9WER
6	9IOS
7	9DXP
	9DSQ
	9DLG

CHAPTER 12: EXECUTION ERROR MONITOR

The Execution Error Monitor, EXEM, provides error messages and, where appropriate, operator options for the standard errors which may occur in FORTRAN object level input/output routines. These standard errors are described in Chapter 11. When loading is under Monitor control, EXEM monitors these errors. However, when object programs are loaded with the BSS loader, an indicative HPR in the storage register accompanies a machine stop under the control of EXE.

EXEM, after diagnosing all standard errors except an end of file in reading tape, transfers control to the appropriate Error Record - that is, the Source Error Record or Machine Error Record - and a message is printed and execution terminated. The option to retry the job is given for machine errors. However, in the case of an end of file in reading tape, EXEM provides the operator with the option to continue execution.

If the library routines that are required for execution are requested during compilation (Sense Switch 5 is DOWN or the LIBE card is in the source deck), the library routine EXE is punched. If subsequently the resulting object programs, including these library routines, are loaded under Monitor control, BSS control will recognize that EXEM is required, rather than EXE, and search the library for EXEM.

The procedure for end of tape while writing is as follows:

The full tape is backspaced over the record whose writing detected the end of tape, and end of file is written and the tape is rewound. A message is printed requesting that the tape which turned on the end-of-tape indicator be changed. When the new tape is ready, the rewound tape dialed off and the Start key depressed, execution resumes by writing on the new tape the record which caused the end-of-tape indicator to be turned on.

The procedures followed by the EXEM for the 16 standard errors which may occur in FORTRAN object level input/output routines are:

I/O Routine	Halt	Reason for Halt	
IOH	HPR 0, 1	Illegal control character in FORMAT statement.	Write off-line: ILLEGAL CHARACTER IN FORMAT STATEMENT. Print on-line and write off-line: EXECUTION TERMINATED BY EXEM. Transfer control to Sign-on.
IOH	HPR 1, 1 HPR 2, 1 HPR 3, 1 HPR 4, 1 HPR 5, 1	Illegal character in data field. Illegal character in octal input data.	Write off-line: Input buffer which contains data which caused stop. ILLEGAL CHARACTER IN DATA ABOVE OR BAD FORMAT. Print on-line and write off-line: EXECUTION TERMINATED BY EXEM. Transfer control to Sign-on.

I/O Routine	Halt	Reason for Halt	
CSH	HPR 0, 2	Illegal data card character.	Write off-line: ILLEGAL CHARACTER ON DATA CARD. Print on-line and write off-line: EXECUTION TERMINATED BY EXEM. Transfer control to Sign-on.
CSH	HPR 1, 2	EOF at card reader.	Write off-line: END OF FILE AT CARD READER. Print on-line and write off-line: EXECUTION TERMINATED BY EXEM. Transfer control to Sign-on.
RER	HPR 0, 3	I/O check light turned on by last read instruction.	Print on-line and write off-line: I/O CHECK LIGHT TURNED ON BY LAST READ INSTRUCTION ON TAPE XX. EXECUTION TERMINATED BY EXEM. Print on-line options: TO RETRY THIS JOB, PUSH START. TO GO TO NEXT JOB, DEPRESS SIGN KEY AND PUSH START.
RER	HPR 1, 3	Tenth consecutive failure in reading tape.	Print on-line and write off-line: REDUNDANCY ENCOUNTERED IN READING TAPE XX. EXECUTION TERMINATED BY EXEM. Print on-line options: TO RETRY THIS JOB, PUSH START. TO GO TO NEXT JOB, DEPRESS SIGN KEY AND PUSH START.
RER	HPR 2, 3	EOF in reading tape.	Print on-line and write off-line: END OF FILE TAPE XX. Print on-line options: TO CONTINUE THIS JOB, PUSH START. TO GO TO NEXT JOB, DEPRESS SIGN KEY AND PUSH START.
TSB	HPR 1, 4	Physical record size exceeds buffer size.	Write off-line: PHYSICAL RECORD SIZE EXCEEDS BUFFER SIZE. Print on-line and write off-line: EXECUTION TERMINATED BY EXEM. Transfer control to Sign-on.
TSB	HPR 2, 4	List exceeds logical record length.	*Write off-line: LIST EXCEEDS LOGICAL RECORD LENGTH. Print on-line and write off-line: EXECUTION TERMINATED BY EXEM. Transfer control to Sign-on.
WER	HPR 0, 5	I/O check light turned on by last write instruction.	Print on-line and write off-line: I/O CHECK LIGHT TURNED ON BY LAST WRITE INSTRUCTION ON TAPE XX. EXECUTION TERMINATED BY EXEM. Print on-line options: TO RETRY THIS JOB, PUSH START. TO GO TO NEXT JOB, DEPRESS SIGN KEY AND PUSH START.
WER	HPR 1, 5	Fifth consecutive failure in writing tape after erasing.	Print on-line and write off-line: REDUNDANCY WRITING TAPE XX. EXECUTION TERMINATED BY EXEM.
WER	HPR 2, 5	Failure in erasing tape.	Print on-line options: TO RETRY THIS JOB, PUSH START. TO GO TO NEXT JOB, DEPRESS SIGN KEY AND PUSH START.
WER	HPR 3, 5	End of tape encountered while writing tape.	Print on-line: END OF TAPE WRITING XX, CHANGE AND PUSH START. The full tape is backspaced, an EOF is written and the tape is rewound. Execution resumes with the record which hit the EOT being written on the new tape.
IOS	HPR 0, 6	Unit address not found in Unit table.	Write off-line: UNIT ADDRESS NOT FOUND IN IOU FOR LOGICAL UNIT XX. Print on-line and write off-line: EXECUTION TERMINATED BY EXEM. Transfer control to Sign-on.
DXP	HPR 1, 7	Exponent exceeds range of DEXP. Permitted range for exponent is x < 88.028.	Write off-line: EXPONENT EXCEEDS RANGE OF DEXP. Print on-line and write off-line: EXECUTION TERMINATED BY EXEM. Transfer control to Sign-on.
DSQ	HPR 2, 7	Negative argument illegal for DSQRT.	Write off-line: NEGATIVE ARGUMENT ILLEGAL FOR DSQRT. Print on-line and write off-line:

*EXEM may be reassembled to permit the storing of zeros in the remaining list items by changing the transfer for HPR (2, 4) stop to PZE CONT

I/O Routine	Halt	Reason for Halt	
			EXECUTION TERMINATED BY EXEM. Transfer control to Sign-on.
DLG	HPR 3, 7	Zero or negative argument illegal for DLOG.	Write off-line: ZERO OR NEGATIVE ARGUMENT ILLEGAL FOR DLOG. Print on-line and write off-line: EXECUTION TERMINATED BY EXEM. Transfer control to Sign-on.

The input/output routine, in which any of the above stops are encountered, can be recognized by the tag of the HPR, as follows:

Tag	Routine Labeled
1	9IOH
2	9CSH
3	9RER
4	9TSB
5	9WER
6	9IOS
7	9DXP
	9DSQ
	9DLG

If an installation wishes to change the procedure for any given error condition, EXEM may be reassembled with the following changes. The address portion of the relevant entry in the Transfer table (starting in symbolic location CODE) may be changed and additional programming corresponding to this change inserted. For example, if a dump is desired whenever an HPR 0, 6 error is encountered, the appropriate entry in CODE may be changed from PZE NOUNIT to PZE HPR 0, 6 and the instruction

```
    HPR 0, 6    CALL    DUMP
```

may be inserted before BUFSIZ EQU 22. The subsequent reassembly will put DUMP in the transfer vector of EXEM and the library subroutine 9DMP will be loaded with the program (in addition to all other required routines). Then, in the case of an HPR 0, 6 error, EXEM will be entered and will call the Dump subroutine. A dump will be given and control will transfer to Sign-on. Note that since the messages for all error conditions except EOF reading and EOT writing are located in Records 10 or 11, the appropriate message will not be given if EXEM is changed in this way. However, the appropriate error code would appear in the sense indicators in the dump (assuming that the additional programming inserted did not destroy the contents of the sense indicators).

For each error stop HPR (A, T), the sense indicators contain an entry to EXEM

```
    A, T, RETURN
```

where RETURN is the absolute address of the point in the I/O subroutine with which execution would resume if the error condition which caused the stop could be corrected. Thus, in the example above, the sense indicators in the dump would contain

```
    0, 6, RETURN.
```

EXEM is also used for source errors arising in Double-Precision Complex Arithmetic Package. The messages for these errors are located in Source Error Record 11. Execution is terminated in all these cases.

9XEM occupies 610_8 locations.

CHAPTER 13: MONITOR CONTROL CARDS AND UTILITY CARDS

Monitor Control Cards

These cards have an "*" in column 1, and, with the exception of the I. D. card, the mnemonics are in columns 7-72. The I. D. card may contain information in columns 2-72.

Card and Mnemonic	Position in Deck	Function
DATE	Optional. If present, must precede I. D. card and/or follow the Monitor START card.	Provides date for printout of page title.
I. D.	The first card in a job, except if DATE card is present.	Used for identifying and accounting information. Signals beginning of a job.
XEQ	Must follow the I. D. card of a job to be executed.	Instructs the Monitor that this job is to be executed.
IOP	Optional	Signals to Monitor that the I/O Package should remain in core with the object program.
DATA	Must follow the physically last program of a job to be executed and precedes the data, if any, for that job.	Signals to Monitor that no more programs in this job follow on the input tape.
CHAIN(R, T)	Must precede the physically first program of each Chain link.	Identifies the Chain link by record number, R, and tape, T.
END TAPE	Must follow the last file on the input tape.	Signals end of input.
LIBE	Must immediately precede source program to which pertinent.	Instructs the Monitor to search the FORTRAN library for subroutines during compilation and to include these with the object program.
LIST; LIST8	Must immediately precede source program to which pertinent.	Instructs the Monitor to list the object program in FAP-type language on the output tape, in two or three column format.

Card and Mnemonic	Position in Deck	Function
CARDS ROW	Must immediately precede source program to which pertinent.	Instructs the Monitor to punch on-line standard FORTRAN relocatable row binary cards, preceded by a BSS loader if a main program.
CARDS COLUMN	Must immediately precede source program to which pertinent.	Instructs the Monitor to punch on-line column binary relocatable cards (no loader) and not to stack the binary output on tape B4 for peripheral punching.
DEBUG	Must immediately follow the last source program, if any, and precede the debug cards for each job or link of a Chain job.	Instructs the Monitor to prepare to read and process debug cards.
LABEL	Must immediately precede source program to which pertinent.	Instructs the Monitor to label off-line card output in accordance with information on appropriate input card.
PACK	Must immediately precede the source program to which it applies.	Instructs the Monitor to pack records on the off-line listing tape.
PRINT	Must immediately precede the source program to which it applies.	Instructs the Monitor to print on-line all the information contained on the listing tape.
ROW	Must immediately precede the source program to which is applies.	Instructs the Monitor to stack relocatable row binary cards on the peripheral punch tape.
SYMBOL TABLE	Must immediately precede source program to which pertinent.	Instructs the Monitor to punch a symbol table.
FAP	Must immediately precede FAP cards and follow control cards for the FAP program.	Instructs Monitor to perform FAP assembly on the following FAP program.
PAUSE	Optional.	Causes a machine halt during reading of input tape.

Utility Cards

START and DUMP Cards

1. START. This is a single self-loading binary card required to initiate Monitor processing. The START card rewinds the input tape only when the START card is followed by the input deck.
2. DUMP. This card causes the Monitor to call the Dump program (Record 2) to dump all of core storage in octal. The following keys are interrogated:

   Sign key:

UP	After dumping, proceed to next job on input tape.
DOWN	After dumping, continue processing current job on input tape.

   Key 1:

UP	No mnemonics on dump.
DOWN	Mnemonics.

   The contents of locations 0, 1, and 2 are destroyed by dumping.

RESTART Cards. These cards permit a continuation of Monitor processing when it is stopped at other than standard diagnostic stops.

1. RESTART THIS JOB. This is a single self-loading binary card. This card repositions the input tape at the beginning of the current job and initiates Monitor processing.
2. RESTART nth JOB. This is a single self-loading binary card. This card positions the input tape at the beginning of the nth job (where n is entered in the address of the keys) and initiates Monitor processing.
3. RESTART BY GOING ON TO NEXT JOB. The START card performs this function.
4. CONTINUE. This is a single self-loading binary card. It initiates Monitor processing at the next program of this job.

PART 2: 7090/7094 FORTRAN II OPERATING UNDER THE 7090/7094 BASIC MONITOR (IBSYS)

Introduction

Part 2 of this manual contains information concerning the operation of the FORTRAN II Processor under the 7090/7094 Basic Monitor (IBSYS).

The FORTRAN II Processor under the Basic Monitor operates in the same manner as the FORTRAN Monitor System operates in the independent version 7090/7094 FORTRAN II discussed in Part 1; compilations, FORTRAN Assembly Program (FAP) assemblies, and binary object programs from previous compilations or assemblies may be executed as parts of a single job. FORTRAN II jobs may also be stacked as input along with jobs for other processors operating under the Basic Monitor.

The FORTRAN II Processor may reside on tape or disk. Substantial savings in setup time are achieved when the Processor operates under the Basic Monitor and resides on disk.

Knowledge of the contents of the IBM 7090/7094 Operating Systems: Basic Monitor (IBSYS) reference manual, Form C28-6248 is assumed.

Basic Monitor Control Cards

All Basic Monitor Control Cards must have a $ in column 1. The specific control instruction is written in columns 2-6. The following is a list of all Basic Monitor Control Cards recognized by the FORTRAN Monitor, and the general action taken when each is encountered. For further information, see the section "Sign-On Record."

1. $EXECUTE FORTRAN -- When this card is recognized by IBSYS, it causes FORTRAN to receive control; when it is recognized by FORTRAN, it sets the FORTRAN Monitor mode of the Monitor.
2. $EXECUTE IBSFAP -- When this card is recognized by IBSYS, it causes FORTRAN to receive control; when it is recognized by FORTRAN, it sets the IBSFAP mode of the Monitor.
3. $JOB -- This card defines an IBSYS job; it is not to be confused with $ID, which identifies a FORTRAN job.
4. $ID -- This card or an *-type I. D. card begins a FORTRAN job deck and may be preceded only by IBSYS control cards or a FORTRAN DATE card.
5. $STOP -- This card signifies the end of a job stack.
6. $IBSYS or $any other text -- This card causes control to be returned to IBSYS.

The following is an example of an input deck setup for the FORTRAN Processor operating under IBSYS:

Column 1	Column 16	Comment
$DATE		IBSYS control cards, as necessary.
$ATTACH		
...		
$JOB		
$EXECUTE	FORTRAN	(This may be IBSFAP, depending on the nature of the job.)
(end of file)		
$ID		(This may be *-type I. D. card.)
...		FORTRAN Monitor control cards and job deck as described in Part I.
...		
...		
(end of file)		
$JOB		
$EXECUTE	FORTRAN	
(end of file)		
*	DATE	(Optional - used for this FORTRAN job only.)
*	any text	(Or $ID.)
...		FORTRAN job deck No. 2.
...		
...		
(end of file)		
$EXECUTE	FORTRAN	(Optional.)
$ID		
...		FORTRAN job deck No. 3. (This job is dependent on the successful completion of job No. 2 since it lacks a $JOB card. Its execution will be prevented if job No. 2 failed.)
...		
...		
...		
...		
(end of file)		
$IBSYS		
...		Any series of jobs for any components of the IBSYS Basic Monitor.
...		
$STOP		End of job stack.

CHAPTER 14: OPERATING INFORMATION

In order to provide uninterrupted operation, all jobs must operate under control of the Basic Monitor. The FORTRAN single-compile mode of operation is not available.

The Basic Monitor Nucleus (IBNUC) and Trap Supervisor (IOEX) will remain in core storage at all times. The origin of the FORTRAN Processor and of the object programs will, therefore, be approximately 2000 (decimal).

Starting, restarting, and dumping procedures are handled by the Basic Monitor. The utility cards for these procedures, supplied with the independent 7090/7094 FORTRAN II System are consequently not used when the FORTRAN Processor operates under IBSYS. The reader is referred to the Basic Monitor (IBSYS) reference manual for information regarding these procedures.

The SYSNAM of the FORTRAN Processor is FORTRA. The $EXECUTE control card may, if desired, refer to FORTRAN, since the terminal N will have no effect.

The SYSCOR location in IBNUC may not have a SYSEND value of less than 77700 (octal). The SYSORG cannot be adjusted unless the FORTRAN Processor is reassembled. The Processor will use all available space in core storage above IOEX except for the top 100 (octal) locations which may be used by installation accounting routines, if desired.

When FORTRAN object programs are in execution, location 2 will not contain the standard IBSYS transfer instruction. Care should be exercised before executing an STR instruction. Location 2 will be restored by the FORTRAN Processor after execution has terminated.

FORTRAN Processor Input and Output

Processor input and output is performed by means of the FORTRAN Input/Output Package (IOP) which in turn utilizes IOEX. See the bulletin IBM 709/7090 Programming Systems: FORTRAN - Input/Output Package for 32K Version, Form J28-6190. It should be noted that the correspondence between logical tape designations used in FORTRAN source program input/output statements and the actual tape assignments at object time is set in the Unit table (IOU) in the FORTRAN Library. The correspondence is not set in the IOP Unit table.

Input to the Processor must be on tape; no provision is made for reading cards on-line. Except for the IBSYS features described under "Control Cards" above, the input deck must be prepared as indicated in the FORTRAN Monitor Section of Part 1. The input for a FORTRAN job must be separated from other input, i.e., there must be an end of file following each FORTRAN job.

Processor output consists of BCD output on the listing tape and binary output on the peripheral punch tape. The binary output for each job occupies one file and is preceded by a separate file containing the I. D. card for that job. End of file is not written on the punch tape when the Processor is in the assemble-only (IBSFAP) mode. The standard output options include the effect of both the PACK and LABEL control cards; therefore, these are unnecessary.

For a Chain job the I. D. card occupies the first file and is followed by a separate file for the binary output of each Chain link.

Tape Unit Assignment

The IOP Unit table is used to assign FORTRAN Processor logical units to Basic Monitor SYSUNI functions.

Function	FORTRAN Logical Unit Number	SYSUNI Function
System	1	SYSLB1
Intermediate	2	SYSUT3
Intermediate	3	SYSUT4
Intermediate	4	SYSUT1
Input	5	SYSIN1
Listing Output	6	SYSOU1
Punch Output	7	SYSPP1
Intermediate (required only for Chain jobs)	8	SYSUT2
(required only if needed for FAP update jobs)	9	SYSCK1
	10	SYSCK2
	11	SYSUAV, 0, 1
	12	SYSUAV, 1, 1
	13	SYSUAV, 0, 2
	14	SYSUAV, 1, 2
	15	SYSUAV, 0, 3
	16	SYSUAV, 1, 3

Logical units 1 through 7 are the only ones necessary for the normal FORTRAN job. Logical unit 8 is necessary only for a Chain job or for updating, and 9 through 16, only if needed for FAP updating jobs.

For maximum efficiency on a two-channel machine, it is suggested that SYSUNI assignments in IBSYS should be such that logical units 1, 4, 5, 6, and any used as update input tapes be connected to one channel, while 2, 3, 7, and 8 be connected to the other.

Logical unit 1 (SYSLB1) may be assigned to a disk file. All other units must be 729 tape drives.

When not used for a Chain job, logical units 8 through 16 are available for FAP update runs. Assignment of these tapes may be made by means of SYSUNI functions or through the Unit Availability tables (SYSUAV). SYSUAV entries in the IOP Unit table are used as follows:

Address:	The tape function (SYSUAV)
Tag:	The channel number (0 through 7 for channels A through H)
Decrement:	The relative tape number in the Unit Availability table for that channel (1 indicates the first tape available for that channel; 2, the second; etc.)

Any units taken from the availability chain during a FORTRAN job are restored to it upon completion of the job.

In order to assure optimum cross-channel buffering for an update run, it will, of course, be necessary for the programmer to take into consideration the IBSYS assignments made at his installation.

Logical units 1 through 7 are checked for availability at the beginning of each job. If any of these SYSUNI functions have not been assigned, an on-line message will be printed and control returned to IBSYS to skip to the next IBSYS job. The operator may depress Sense Switch 1 and read-in, on-line, the necessary $ATTACH and $AS control cards followed by a $RESTART control card. SYSIN1 will then be backspaced to the current $IBSYS job and processing will resume at that point.

Logical units 8-16 are not checked for availability at the beginning of a job. For example, if one of them is referenced by a Chain job (which requires unit 8) or during a FAP update run, and it is found not to be available when required, an on-line error message will be printed notifying the operator that logical tape N is not in the Unit table. Control will transfer to the FORTRAN Machine Error Record where an operator action pause will occur. The retry option should not be taken at this time, since the job may be successfully rerun only after the necessary SYSUNI assignments have been made.

To alter unit assignments, it is necessary either to change the SYSUNI Table in IBNUC, with IBSYS $ATTACH cards or by reassembly, or to reassemble the IOP Unit Table with the desired tape function names.

The FORTRAN Processor will set density on logical tape units 1-7 according to the SYSUNI entries for those tapes. An installation may, if desired, cause the FORTRAN Processor not to set density by inserting a non zero address in symbolic location TAPNO of the IOP Unit table.

Output Tape Switching

If the FORTRAN Monitor listing tape (SYSOU1) reaches the end of tape, or if permanent redundancies occur during a FORTRAN job, the FORTRAN Input/Output Package (IOP) checks the SYSUNI table for an alternate output unit (SYSOU2). If it is available, IOP will switch the SYSUNI entries for the two units and print on-line notification of the tape alternation. When SYSOU2 is not available, or if it is identical to SYSOU1, a request to mount a new tape is printed. In either instance, an end-of-file mark, a trailer label, and a second end-of-file mark are written on the completed tape and it is unloaded. The record which could not be written on the old tape is written on the new one, and the job is continued.

Tape Error Procedures

In addition to the FORTRAN Input/Output Package error procedures, read or write errors are also handled by IOEX. Messages appear on-line to indicate tape errors found by IOEX as follows:

NOISE RECORD DISCARDED
IOEX has signaled IOP that a noise record has been read. IOP verifies that it is a noise record, the message is printed, and the noise record is ignored.

25 ERASES DURING WRITE
IOEX has rewritten a record and found redundancies twenty-five times. Erasures continue until the record has been successfully written.

NOISE ON ERASE
IOEX, while erasing, has found a bad spot which cannot be removed. This remains as a noise record. Further attempts to write are continued until a record has been written successfully.

I/O CHECK
IOEX has found that the I/O Check Indicator was ON and pauses after printing the message. Since the channel cannot be determined, no indication is given to the FORTRAN Processor. If the operator depresses the Start key, FORTRAN will proceed as if the check has not occurred.

EOT ON ERASE
IOEX has sensed the end-of-tape indicator while erasing. This condition causes an unconditional halt in IBSYS.

CHAPTER 15: THE FORTRAN MONITOR

In general, the FORTRAN Monitor operates under IBSYS as it does in the independent version of FORTRAN II. The sections which follow describe the cases in which the operation of the two systems differs.

Monitor Flag Cells

The first 12 locations from SYSORG to the symbolic location BOTTOM (the origin of FORTRAN) are used as a communications region as follows:

BOTTOM-12 contains a flag when the FORTRAN Processor is operating.

BOTTOM-11 through BOTTOM-5 are used to save machine registers and for communication with the DUMP routine.

However, BOTTOM-4 through BOTTOM-1 are FLAGBX, LINECT, DATEBX, and PRCBRK as they are in the independent version of FORTRAN.

IBSYS Flag Cells

The IBSYS flag cells are:

1. SYSJOB -- Since the FORTRAN Monitor restores the availability chain when it uses it, the "restoration needed" bit in this cell will not cause the Monitor to reload IBSUP between jobs. The bit in SYSJOB that indicates that a previous IBSYS job segment failed will be recognized by Monitor Scan and will prevent execution. FORTRAN will set this bit on for an assembly or compilation error or if execution is terminated by EXEM.
2. SYSCUR -- This cell is used to initially determine the Monitor mode. If SYSCUR contains IBSFAP when FORTRAN receives control, the assemble-only mode is set; if it does not contain IBSFAP, the execution mode is set. Each FORTRAN record loads SYSCUR with its name.

The publication IBM 7090/7094 IBSYS Operating System: System Monitor (IBSYS), Form C28-6248-1, contains further information on the IBSYS flag cells.

FORTRAN Input/Output Package (IOP) Record

IOP is used by the FORTRAN Processor for all input and output functions. It occupies upper memory at all times during a FORTRAN run except when the object program is in execution. Calling sequences to IOP are transformed into calling sequences to IOEX so that IOP is an interface between FORTRAN and IOEX.

FORTRAN System records are loaded through IOP, which uses the SYSLDR routine in IOEX.

The Basic Monitor passes control directly to IOP, which is the first record of the Processor. When this occurs, IOP resets all flag cells. Control is returned to IOP directly after object program execution. When this occurs, IOP recognizes the indicator in BOTTOM-12 and does not reset the flags. Control is then passed from IOP to the System Positioning record which follows it.

System Positioning Record

The card-to-tape simulator which occupies this record in the independent version of FORTRAN is not used under IBSYS.

This record calls the next FORTRAN record (Dump, Sign-On, Machine Error, or Source Error) according to indicator bits in FLAGBX.

The FORTRAN Dump record is used for object program dumps only. It is called by the DUMP (or PDUMP) library routine.

The core storage locations from BOTTOM-11 through BOTTOM-5 are used as a communications region between the DUMP (or PDUMP) library routine and the Dump record.

The contents of core storage is saved for the dump on logical unit 2 (SYSUT3). If the IOP Unit table has been changed so that logical unit 2 is not SYSUT3, the DUMP library subroutine must be changed accordingly.

Sign-On Record

At the beginning of each job, control is passed to the FORTRAN Sign-On record. Sign-On performs the following functions:

1. Ensures that location 2 contains a TTR SYSDMP.
2. Sets the current date from SYSDAT. (This may be reset for the current job with a *DATE card.)
3. Prints the tape statistics and the line count for the preceding job, if necessary.
4. At this point, a test is made for the assemble-only (IBSFAP) mode. If the Monitor is in an IBSFAP mode, Sign-On reads cards from SYSIN1, processing IBSYS control cards (as described in the following text) until a card without a $ in column 1 is encountered. When such a card is encountered, Monitor Scan receives control to process it. In the execute (FORTRAN) mode, processing proceeds as follows:
   a. An end of file is written on SYSPP1 unless it is already positioned at the beginning of a file.
   b. If SYSIN1 is not positioned at an end of file, or if the preceding card on SYSIN1 is neither the first card of a file nor a $-type control card, a file is skipped on SYSIN1.
   c. The FORTRAN job I.D. card is now processed. This card is defined as the next card on SYSIN1, with an * in column 1, that is not a DATE card. More than one card may be read and processed in the search for the I.D. card as long as the cards preceding the I.D. card are either IBSYS control cards or the DATE card. When the I.D. card is encountered, the search for an *-type I.D. card is discontinued. A *END TAPE is treated as an I.D. card and also returns control to IBSYS. If any other card is encountered, the I.D. card is assumed to be missing, and an option is given to the operator to use the following as I.D.:

      * NO I.D. CARD FOR THIS FORTRAN JOB.
   d. The I.D. card is written as a file on SYSPP1.
   e. Sign-On retains control and continues to read SYSIN1 until a card other than an IBSYS control card is encountered. At this point, Monitor Scan receives control to process this card.

The IBSYS control cards recognized by Sign-On and the action taken are as follows:

1. $EXECUTE -- If the operand of the EXECUTE is FORTRAN or IBSFAP, the Monitor is set to that mode. If the operand is anything else, it is stored in the IBSYS communication cell SYSGET and control is returned to IBSYS through SYSRET.

2. $JOB -- The cell SYSGET is loaded with the word "IBSBSR" and SYSRPT is called. If no interrupt occurs, SYSGET is set to "IBSYST" and SYSIDR is called.

3. $ID -- SYSIDR is called. This card is then treated as a FORTRAN I.D. card if there has been no previous one for this job.

4. $STOP -- The cell SYSGET is set with this operation and control returns to IBSYS through SYSRET.

5. $IBSYS or $any other text -- Control is returned to IBSYS through SYSRET.

When control is returned to IBSYS, Sign-On clears the FORTRAN flag in location BOTTOM - 12.

Sign-On prints tape statistics for the previous job on SYSOU1. If an installation wishes to have the statistics appear on-line, the operations in both location GTCOM+1 and STAT16+1 should be changed to SLN 4. To eliminate the statistics entirely, change location HTPSTS+2 to TRA FCTWO. (For elimination of tape statistics from BSS control, see the section "BSS Control Records.")

Sign-On may be programmed by an installation to handle each accounting job. The top 100_8 locations in core storage are not used by the FORTRAN Processor and may be used to save such desired information.

FORTRAN Assembly Program (FAP)

The System Symbol table in FAP contains both FORTRAN and IBSYS symbols.

Logical tapes 8 through 16 may be used for FAP update runs. In a Chain job, however, tape 8 is used as an intermediate tape by the Processor.

BSS Control Records

The BSS Control records load object programs starting from location BOTTOM.

One hundred (octal) locations in upper core storage are reserved for installation use and are not used by the FORTRAN Processor. COMMON is relocated downward by this amount during loading; references to COMMON in object programs will be changed to start from 77361_8 (instead of 77461_8 as compiled). If it is desired to readjust COMMON and not to reserve the top of memory, location COMN should be changed so that it contains zero. Also, the value of TOPMEM, which is the location of the top of memory available for use by the compiler, should be changed in the FAP System Symbol table (see FAP Manual) and the System should be reassembled. To accomplish this for a single job only, a control card, when punched as follows, may be used preceding any other binary cards in the job deck:

Column 1: rows 11, 7, and 9 punched
Column 2: row 12 punched
Remainder of card blank

Tape statistics are written both on-line and off-line before control is passed to the object program. In order to delete the statistics, change location TOPR1 as follows:

TRA	TOPR2 -	to delete tape statistics
STL	NOSTON -	to delete statistics on-line only
STL	NOSTOF -	to delete statistics off-line only

Library Search Records

The BSS Control Library Search records read blocked library card images from the library file of the FORTRAN Processor. The library will be searched from a unit other than logical unit 1 if the decrement of location (LIBT) in IOP is changed to contain the unit number. However, if the unit designation is not 1, it must be greater than 8.

CHAPTER 16: THE FORTRAN COMPILER

FORTRAN II source programs written for the independent version of FORTRAN II need not be changed to operate under the Basic Monitor. Some extremely large, complicated programs may not compile, however, because the origin of the Processor is higher than the origin of the independent version of FORTRAN and because table space is approximately 2000 (decimal) locations less under IBSYS. With this exception, input acceptable to the FORTRAN Compiler under IBSYS is also acceptable to the independent version of FORTRAN and vice versa.

Both the FORTRAN Compiler under IBSYS and the independent version of FORTRAN compile object programs (including calling sequences to library routines) in the same manner.

The FORTRAN Compiler under IBSYS provides an option to produce row binary cards and the BSS Loader as output. It will not provide row binary library routines. If row binary library routines are requested, the request is ignored and a message is printed to that effect.

Section Six of the compiler will search the library on a unit other than logical unit 1 if the decrement of location (LIBT) in IOP is changed. For a description of this, see the section on Library Search.

CHAPTER 17: OBJECT PROGRAM EXECUTION

Except for library routines, column binary decks produced by the independent version of FORTRAN may be used with IBSYS. Library routines contained in the independent version of FORTRAN are not compatible with the Basic Monitor. If they are included in such decks, they must be discarded.

Chain links may be stacked on logical tape units 2, 3, and 4. No change is necessary to source or object programs written for or compiled by the independent version of FORTRAN when they are used in connection with IBSYS. The chain tapes used will, however, not necessarily be those specified in the source program. Tapes specified as B2 will be assigned to logical unit 2 (SYSUT3); tapes specified as B3 will be assigned to logical unit 3 (SYSUT4); tapes specified as A4 will be assigned to logical unit 4 (SYSUT1). Thus, the actual tape unit for Chain links will depend on the SYSUNI assignments and the IOP Unit table. Note that this assignment is independent of the object program IOU table. If the SYSUNI designation for logical units used as chain tapes (2,3,4) is changed in the IOP Unit table, the CHAIN subroutine must be changed accordingly.

FORTRAN object programs in execution may destroy location 2. A transfer to SYSDMP is stored there upon completion of execution.

Object programs are loaded into core storage above IOEX. Caution should be exercised in FAP coded routines so that IOEX is not destroyed. It must remain in core storage at all times for proper operation of the Basic Monitor. In a FAP program using IOEX, the programmer should make sure that the proper channel traps are enabled when necessary since FORTRAN object program input/output library subroutines disable all channel traps. UCB words should be updated in FAP routines which perform input/output operations independently of IOEX so that IOEX can operate successfully for the succeeding job.

CHAPTER 18: THE FORTRAN LIBRARY

Library routines supplied with the FORTRAN Processor operating under the Basic Monitor must be loaded and executed under FORTRAN Monitor and Basic Monitor control. There is no provision made for execution independent of the Basic Monitor. Since library routines supplied with the independent version of FORTRAN are not compatible with IBSYS, they in turn may not be used with the FORTRAN Processor under IBSYS.

No provision is made to punch row binary cards from the library. However, for object programs which are too large or otherwise incompatible with the Basic Monitor, row binary cards may be produced on option. These may be used for loading with the BSS loader along with library routines supplied with the independent version of FORTRAN.

The records in the library file are named 9FL001, 002, etc. They contain 12 relocatable binary card images and the necessary control words to conform with the IBSYS System record format. The last record is a special end-of-file indicator that is used when the library resides on disk; it is ignored for a tape system.

FORTRAN Monitor Subroutines

The EXIT subroutine uses IOEX to call the FORTRAN Processor. It sets the flag indicating that control should transfer to Sign-On and then calls in IOP.

Two additional entry points in EXIT are used by the other Monitor library subroutines: EXITFN, which calls in IOP, and EXSEL, which is an IOEX select routine.

The DUMP (or PDUMP) subroutine uses IOEX to save core storage before reading in IOP and the Dump record. The upper section of core storage, which will be occupied by IOP and the Dump record, is saved on SYSUT3. If the IOP Unit table has been changed so that logical unit 2 does not correspond to SYSUT3, the DUMP subroutine must be changed accordingly. Lower memory, including IOEX, is dumped as of the time the Dump record is actually operating. It is not dumped as it existed when the object program called DUMP or PDUMP.

The CHAIN subroutine uses IOEX for reading links from tape. It does not use the object program (IOU) table. It reads links from SYSUNI tapes corresponding to the IOP Unit table of the Processor. If the IOP Unit table assignment is changed for units used as Chain links, a corresponding change must be made in the CHAIN subroutine.

The Execution Error Monitor (EXEM) subroutine is loaded, and all object program references to (EXE) are changed to (EXEM) when loading by the BSS Control record of the FORTRAN Monitor. (EXEM) handles two IOS subroutine error conditions in addition to those handled by the independent version of FORTRAN: SYSUNI function not assigned (code 1, 6) and disk or other illegal unit assigned (code 2,6). In both of these cases, (EXEM) will terminate execution and transfer control to the Machine Error record.

Input/Output Library

The FORTRAN I/O library subroutines used by object programs do not make direct use of IOEX. To permit uninterrupted operation of IBSYS, they keep counts in the Unit Control Blocks updated. The subroutines provide facilities for 729 tape drives, on-line printer, card reader, and card punch; there is no provision made for disk files.

Channel traps are disabled when the flow of the object program enters the Input/Output Supervisor (IOS) subroutine. (IOS) initializes instructions for the I/O unit specified in the SYSUNI or SYSUAV tables which correspond to the (IOU) logical unit number. An entry point in (IOS), (UCB), is set to contain the address of the first word of the Unit Control Block pertaining to the unit being used. Error conditions which cause transfer to (EXE) are as follows: (IOU) table entry not found (code 0,6); SYSUNI or SYSUAV function not assigned (code 1,6); and illegal unit assigned (code 2,6). (IOS) causes a transfer to EXIT if an end of file is read on the input tape defined by an MZE in the (IOU) table (logical tape 6, SYSIN1 in the table as distributed).

The Input/Output Channel-Unit table (IOU) for FORTRAN object programs assigns logical units to SYSUNI functions as follows:

FORTRAN Unit Number	SYSUNI Function
1	SYSLB1
2	SYSUT3
3	SYSUT4
4	SYSUT1
5	SYSIN1
6	SYSOU1
7	SYSPP1
8	SYSUT2

Table A shows the symbolic cards which will produce the relocatable binary cards in the FORTRAN library as distributed.

All entries in the (IOU) table contain an address in the System Unit Function table, SYSUNI, or the Unit Availability table, SYSUAV, which are found in the Nucleus. The physical unit addresses are specified in the Unit Control Blocks in IBNUC.

The format of the (IOU) table is as follows:

(IOU)-3	contains the address of the system printer entry in SYSUNI
(IOU)-2	contains the address of the system punch in SYSUNI
(IOU)-1	contains the address of the system card reader entry in SYSUNI
(IOU)	contains the number of table entries following this location.

The next N locations correspond to the logical tape numbers from 1 through N, and contain the address of a SYSUNI or SYSUAV table entry. The decrement of locations pointing to the SYSUAV table specifies the relative unit to be used in the availability chain. For example, a 2 in the decrement field specifies the 2nd available unit. A negative sign bit defines the Monitor input unit. An end of file while reading that unit will cause the EXIT subroutine to be called.

```
1       8       16
        IOU TABLE AS FOUND IN THE FORTRAN LIBRARY
        FAP
        COUNT   18
        SST
        LBL     9IOU, X
        ENTRY   (IOU)
        PZE     SYSPRT    SYSTEM PRINTER
        PZE     SYSPCH    SYSTEM PUNCH
        PZE     SYSCRD    SYSTEM CARD READER
(IOU)   PZE     NTAPES
        PZE     SYSLB1    LOGICAL UNIT 1 (SYSTEM)
        PZE     SYSUT3                 2 (UTILITY)
        PZE     SYSUT4                 3 (UTILITY)
        PZE     SYSUT1                 4 (UTILITY)
        MZE     SYSIN1                 5 (INPUT)
        PZE     SYSOU1                 6 (OUTPUT)
        PZE     SYSPP1                 7 (PUNCH)
        PZE     SYSUT2                 8 (UTILITY)
*       INSERTIONS AND DELETIONS SHOULD BE MADE
*       BETWEEN NTAPES AND (IOU)
NTAPES  EQU     *-(IOU)-1
        END
```

TABLE A.

To change the IOU assignment of units, the following procedure is necessary.

1. Change the appropriate cards in a symbolic deck for (IOU). Table B, below, indicates this procedure by an example for 12 units, some utilizing tapes in the Unit Availability tables (SYSUAV).
2. Assemble this symbolic deck using FAP.
3. Replace the two cards labeled 9IOU0000 and 9IOU0001 in the FORTRAN Library with the cards produced from the FAP assembly.
4. Generate a new system using this library.

```
1       8       16
*       EXAMPLE OF IOU TABLE WITH 12 UNITS
*       FAP
        COUNT 22
        SST
        LBL     9IOU,X
        ENTRY   (IOU)
        PZE     SYSPRT    SYSTEM PRINTER
        PZE     SYSPCH    SYSTEM PUNCH
        PZE     SYSCRD    SYSTEM CARD READER
(IOU)   PZE     NTAPES
        PZE     SYSLB1    LOGICAL UNIT 1 (SYSTEM)
        MZE     SYSIN1                 2 (INPUT)
        PZE     SYSOU1                 3 (OUTPUT)
        PZE     SYSUT1                 4 (UTILITY)
        PZE     SYSUT2                 5 (UTILITY)
        PZE     SYSUT3                 6 (UTILITY)
        PZE     SYSUT4                 7 (UTILITY)
        PZE     SYSPP1                 8 (PUNCH)
        PZE     SYSCK1                 9 (CHECK OR
                                         UTILITY)
        PZE     SYSCK2                10 (CHECK OR
                                         UTILITY)
```

```
         PZE       SYSUAV,,1          11 (UTILITY -
                                         1ST IN
                                         AVAILABILITY
                                         CHAIN)
         PZE       SYSUAV,,2          12 (UTILITY -
                                         2ND IN
                                         AVAILABILITY
                                         CHAIN)
*        INSERTIONS AND DELETIONS SHOULD BE MADE
*        BETWEEN NTAPES AND (IOU)
NTAPES   EQU       *-(IOU)-1
         END
```

TABLE B.

CHAPTER 19: MAINTAINING THE FORTRAN PROCESSOR

The FORTRAN Processor under IBSYS is edited by means of the System Editor (IBEDT). For a description of the editing procedure, see the IBSYS reference manual. In order to update the FORTRAN Library, however, a blocked library tape must first be prepared by the FORTRAN Library Editor.

FORTRAN Library Editor

The Library Editor record reads binary card images from SYSIN1 and prepares blocked records for the FORTRAN Library file on SYSUT4. This file on SYSUT4 may then be copied onto the System by IBEDT.

The FORTRAN Monitor control card

```
* EDIT LIBE
```

causes control to be passed to the Library Editor. The Library Editor reads the binary card images from SYSIN1 and prepares blocked records which are written on SYSUT4. An end of file on SYSIN1 signals the end of the input. When this end of file is encountered, a trailer label is written on SYSUT4. The trailer label signals an end of library on disk and is ignored on tape. Also, an end of file is written on SYSUT4 and control is returned to the Sign-On record.

Editing with IBEDT

To update the FORTRAN Processor, including the Library file, the following card sequence may be used:

```
1         7         16

$REWIND             SYSUT4
$EXECUTE            FORTRA
$ID
*         EDIT LIBE
            .
          (column binary library routines)
            .
(end of file)
$IBSYS
$REWIND             SYSUT4
$IBEDT
                    *EDIT
                    *MODIFY        9F0300
                      .
                    (column binary modifications)
                      .
                    *MODIFY        9F2100
                      .
                      .
FILE                *REPLACE       9FL001,SYSUT4
                      .
(end of file)
```

FORTRAN Processor Records

The FORTRAN Processor consists of six files on the IBSYS System tape. File 1 contains a loader which defines the assemble-only mode; file 2 contains the FORTRAN Monitor and FAP records; file 3 contains the FORTRAN Compiler records and duplicates of the last three Monitor records; file 4 contains the blocked library records; file 5 contains the General Diagnostic record; and file 6 contains the FORTRAN Library Editor record.

The arrangement of records in each file and the label which is loaded into SYSCUR for each record is shown in columns 1 and 2 of the table below. The third column shows the serialization on the symbolic cards and listings for each record of the Processor.

Record	Label	Serialization
IBSFAP loader	IBSFAP	9IBS
(end of file)		
IOP	FORTRA	F00
System Positioning	9F0100	F0A
Dump	9F0200	F0B
Sign-On	9F0300	F0C
FAP	9F0400, 9F0500	F0D, F0E
Scan	9F0600	F0F
Debug	9F0700	F0G
BSS Control	9F0800	F0H
Library Search	9F0900	F0I
Machine Error	9F1000	F0J
Source Error	9F1100	F0K
Dummy	9F1200	F0L
(end of file)		
Section 1	9F1300-9F1700	F1A-F1E
Section 2	9F1800-9F2100	F2A-F2D
Section 3	9F2200	F3A
Section 4	9F2300-9F2500	F4A-F4C
Section 5	9F2600-9F2900	F5A-F5D
Section 6	9F3000, 9F3100	F6A, F6B
Debug	9F3200	see file 1
BSS Control	9F3300	see file 1
Library Search	9F3400	see file 1
(end of file)		
Library	9FL001,9FL002, etc.	L
(end of file)		
General Diagnostic	9D0000	XGD
(end of file)		
Library Editor	9LEDIT	ZLED
(end of file)		

APPENDIX A: SOURCE LANGUAGE DEBUGGING AT OBJECT TIME

The FORTRAN II Monitor includes facilities for specifying object-time storage dumps in FORTRAN source program-type language. These dumps are dynamic; they are obtained by interrupting execution and collecting the desired information, after which control is returned to the object program. Provision is made for dumping during execution of any main program or subprogram as long as execution is under Monitor control and the object deck contains a Symbol table.

Symbol Table

A separate SYMBOL TABLE card is required for each main program or subprogram in which dumping is desired at object time. The Symbol table contains information regarding the relative location of all variables and numbered statements in the source program.

If a Symbol Table is not present for a program, all requests for dumps during execution of that program will be ignored (although the debug cards will be listed on the output tape).

Debug Cards

These cards become part of the job deck. They may be altered at object program load time, thereby permitting selective dumping without recompiling. The debug cards for a job or link must be preceded by a DEBUG control card. This packet must follow the last source program (if any) and must precede the first binary program (if any) of each job or Chain link.

Name Card

Format:	Column 1	N
	Columns 7-72	Subprogram Name

The main program is indicated by the absence of a name on the Name card. All DUMP cards following a Name card refer to the named program. The packets of Name cards and their related DUMP cards may be in any order.

DUMP Card

Format:	Columns 1-5	Source statement number at which dump is desired.
	Column 6	Continuation (as in FORTRAN source language)
	Columns 7-72	Field 1 \$ Field 2 \$ Field 3

This card is used to stipulate the frequency of dumps, the criteria for determining whether to dump, and the quantities to be dumped. Multiple dumps at the same statement are permitted.

Field 1

Format: DUMP n_1, n_2, n_3
where n_1, n_2, and n_3 are decimal integers.

A dump will occur the n_1th time control passes through the indicated source statement and will occur every n_3th time thereafter until it is about to exceed n_2. The n_3 subfield may be omitted, in which case it is taken to be one.

Field 2

Format: IF ($v_1 \pm v_2$) a_1, a_2, a_3
where v_1 and v_2 are each either a subscripted or nonsubscripted unsigned variable or an unsigned decimal constant.

Each subscript must be expressed as a single unsigned integer as in an EQUIVALENCE statement. The quantity v_2 may not be omitted and must be of the same mode as v_1.

Each a_i is either YES or NO, indicating that a dump should or should not occur, depending on whether $v_1 \pm v_2$ is negative (a_1), zero (a_2), or positive (a_3).

This field provides a method of controlling dumping by testing the values of source variables. It is connected with Field 1 in a logical "and" fashion.

This field may be omitted; that is, the DUMP card may be of the form Field 1 \$ Field 3.

Field 3

Format: List
where List specifies the locations to be dumped and may include any of the following (each item must be separated by a comma):

1. Subscripted or nonsubscripted variables. The mode of the variable is unimportant. Each subscript must be expressed as a single un-

signed integer as in an EQUIVALENCE statement. For example, A(2,2), with dimensions 3 x 3, must be referenced as A(5).

2. Arrays
Format: v(c-c')
where v is an array name of any mode, and c and c' are each unsigned integers; c' must be larger than c.
Each item from v(c) to v(c') will be dumped. Thus, B(2-5) will cause B(2), B(3), B(4), and B(5) to be dumped.

3. Common Data
Format: COMMON DATA
or COMMON DATA (s_1-s_2)
where s_1 and s_2 are unsigned integers and s_2 is greater than s_1, or s_1 and s_2 are variable names and s_1 and s_2 are single-word source program variables that appear in common. The entire COMMON region or the range of COMMON from the s_1th location through the s_2th location of COMMON are dumped. The output of this dump is identified by octal location rather than variable name.
Example: If Field 3 is coded as COMMON DATA (1-25), then, whenever the conditions for dumping are met, locations 77431_8 through 77461_8 will be dumped.
If Field 3 is coded as COMMON DATA (A-B), then that portion of COMMON from the location containing A to the location containing B will be dumped.

4. Program Data
Format: PROGRAM DATA
or PROGRAM DATA (s_1-s_2)
This is the same as 3, but pertains to source variables that do not appear in COMMON statements.

5. Erasable Data
Format: ERASABLE
or ERASABLE (s_1-s_2)
This is the same as 3, but pertains only to erasable storage, which is a small block of locations below the lower limit of program data.

6. Octal Dump
Format: OCTAL DUMP
or OCTAL DUMP (v_1-v_2)
where v_1 and v_2 are each either source statement numbers or subscripted or nonsubscripted unsigned variables or COMMON DATA (s_1), or PROGRAM DATA (s_1), or ERASABLE (s_1). Subscripts must be one-dimensional and must be unsigned integers. v_2 must denote a higher location than v_1. It is also permissible to use a mixed form such as COMMON DATA (s_1) - PROGRAM DATA (s_2).
This causes all of core storage or the range from v_1 up to v_2 to be dumped in octal.

7. Decimal Dump
Format: DECIMAL DUMP
or DECIMAL DUMP (v_1-v_2)
This is the same as 6, but causes all of core storage or the range from v_1 up to v_2 to be converted according to the G specification prior to output.

8. BCD Dump
Format: BCD DUMP
or BCD DUMP (v_1-v_2)
This is the same as 6, but all of core storage or that part of core storage in the range v_1, up to v_2, will be dumped in BCD, i.e., according to the A specification.
Note: If illegal characters are contained in the dump, a BCD dump will not print on a 720 or 717 Printer, but will print on a 1403.

G Conversion

With the exception of octal and BCD dumps (see above) all quantities to be dumped are converted according to the G specification prior to output. G conversion causes the location to be examined, and proceeds as follows:

1. If the prefix, tag, and address are all zero, the location is assumed to contain an integer, and I conversion is used.
2. If the prefix, tag, or address is not zero, the location is assumed to contain a floating point quantity. If the quantity is less than 1/10 or greater than 10d + 1, where d is $\leq$ 9, E conversion is used; otherwise, F conversion is used.

Description of Dumping

Dumps are taken with regard to the execution of source statements as follows:

Statement	When Dump Occurs
Arithmetic	After the statement is evaluated
IF	After the expression is evaluated and before the transfer
GO TO (Unconditional and Computed)	Before the transfer
RETURN	Before the transfer
CONTINUE (at end of a DO)	Before the statement is executed
CONTINUE (not at end of a DO)	May not be dumped
Input/Output	Before the statement is executed
CALL	May not be dumped
DO	May not be dumped
GO TO (Assigned)	May not be dumped

Output is on logical tape 6, intermixed with programmed output, and is identified by subprogram name (if a subprogram), statement number, and a count indicating the number of times (modulo 32,767) control has passed through the statement. Variables and arrays specifically listed on the DUMP card are labeled by name.

Program data, Erasable data, and COMMON data dumps are labeled with respect to the octal locations given in the storage map. Octal dumps are labeled with the octal locations into which the program has been loaded after relocation.

Lines consisting entirely of zeros are omitted. Except for single-word variables and arrays, all information is printed six locations to a line. If the number of items requested is not a multiple of six, the last line will be completed with successive locations from core storage.

After loading a subprogram, a check is made to see if dumps are requested in the subprogram. If so, the program break is automatically extended to include counter tests, calling sequences, and formats for dumping. Instructions required by the Debug program are inserted in the program as required. Arguments in subprograms may not be dumped.

There are no error stops in the Debug program; an attempt is made to salvage as much information as possible from incorrectly punched debug cards.

FORTRAN output routines are used by the debugging package and thus (STH) and (FIL) are required in the object program. The Debug program will cause a library search for these routines if they are not included with the object program.

Job Deck Formats

Both execute and Chain jobs may contain debug cards as follows:

Execute Job

1. I.D. card.
2. XEQ card.
3. FORTRAN source deck(s) with control cards and, if desired, FAP symbolic deck(s) with control cards (although the debug facilities may not be used for FAP).
4. DEBUG card, if desired.
5. Debug cards, if desired.
6. Binary programs, if any; each program to be debugged preceded by its Symbol table.
7. DATA card, if data.
8. Data cards, if any.

Chain Job

1. I. D. card.
2. XEQ card.
3. CHAIN (R, T) card.
4. Set of cards as described by items 3,4,5, and 6 under Execute Job.
5. As many sets of types 3 and 4 above as desired.
6. DATA card, if data.
7. Data cards, if any.

Symbol Table Format

The first card in the Symbol table has the following format:

9L	Prefix:	6	
	Decrement:	Card word count (excluding first two words)	
	Address:	Zero	
9R		Checksum	
8L		Subprogram name in BCD packed to left and filled with blanks (if main program, it will be filled with zeros)	
8R	Prefix:	Zero	
	Decrement:	Number of symbols referred to in symbol table	
	Address:	Zero	
7L, 6L, ...		Source program variable name	in
		Internal symbol for erasable data	BCD
		External Formula Number	
		Packed to left and filled with blanks	
7R, 6R, ...	Prefix:	Relocation bits 010	
	Decrement:	0: Data in common	
		2: Other source program data	
		3: Erasable and program data	
		4: External formula number	
	Address:	Relative address of table entry; ascending order sort on decrement, descending order sort within the decrement value. The second, third, ... cards in the Symbol table have the same format as the first except that 8L is the same as 7L, 6L, ... and 8R is the same as 7R, 6R, ...	

Storage Requirements at Execute Time

If debugging is requested, 60 + (2 times the number of different statements at which dumps are requested) additional storage locations are required. In addition, 26 locations are required for each DUMP card without an IF test, and 32 locations are required for each DUMP card with an IF test. For each item in the List, the following number of additional locations are required.

Each subscripted or nonsubscripted variable	2
Each array	14
Each Erasable storage dump	4
Each COMMON data, Program data, or octal data dump	12

Also, for each item in the list, the following locations are required for format:

Each subscripted or nonsubscripted variable	4
Each array	9
Each Erasable data, COMMON data, Program data, or octal data dump	3

When debugging is requested, the object program will be loaded starting three locations higher than if there were no debugging request.

Table Sizes for Debugging Routines

1. The maximum number of subprograms in which dumps may occur is 20.
2. The maximum number of nonblank characters on DUMP cards per program or subprogram is 1200-2(n + 2), where n is the number of different statements at which dumps have been requested.
3. The maximum number of nonblank characters on all DUMP cards is 3000-2(n + 2), where n is the number of different statements at which dumps have been requested.
4. The maximum number of different statements at which dumps may occur in any one program or subprogram is 10.
5. The maximum number of different statements at which dumps may occur for a program and all its subprogram is 25.
6. The maximum number of Symbol Table entries per program or subprogram is 500.
7. The maximum format length permitted for a single statement is 200 locations.
8. The number of debugging output lines is limited to 1,000. Subsequent lines will not be written.

In order to change this limit, symbolic card LH000390 in the FORTRAN library routine STH should be changed as follows:

```
     TXH   STHX,4,xxxx
```

where xxxx is the maximum number of lines of debug output permitted.

APPENDIX B: STANDARD ERROR PROCEDURE

The FORTRAN standard error procedure permits an error program, called during execution of an object program, to determine the external and internal formula numbers of the source program statement containing the error and, in the case of subprograms, to determine the name of the subprogram.

Activating The Standard Error Procedure

The standard error procedure is activated as follows:

FORTRAN II Operating Under IBSYS

1. Symbolic location SEPFLG in record 4 should contain 777777777777_8.
2. Symbolic location FLTR00 in record 14 should contain the instruction SXD FLTR05,4.

FORTRAN II Independent of IBSYS

Remove cards 9F04FLOW and 9F14FLOW from the Editor deck.

Flow Tracing Instructions

When the standard error procedure has been activated, the compiler will lengthen the calling sequence to the following closed subroutines; arithmetic statement functions; all library functions except 9FPT and the input/output subroutines; and FUNCTION and SUBROUTINE subprograms.

General Subroutine Calling Sequence

The general subroutine calling sequence is as follows:

```
   TSX    NAME,4        NAME is the subroutine
                        name
   TSX    LARG1         location of first argument
   TSX    LARG2         location of second argument
    .      .             .
    .      .             .
   TSX    LARGn         location of nth argument
   (Return point from subroutine)
```

For arithmetic statement functions and library functions, n is zero, since the arguments are stored elsewhere and are processed prior to the transfer to the function. The SUBROUTINE subprogram may have n = 0, i.e., no arguments.

Additional Flow Tracing Instructions

In order to provide the error program with information for the flow trace, the following instructions are added to the general calling sequence:

```
   NTR    *+2,0,A
   PZE    C,0,B
```

where:

1. A is the external formula number (i.e., source program statement number) of the source statement that contains the subroutine call; if this source statement does not have a statement

number, A is the external formula number of the last preceding source statement that has a statement number. If no statement numbers are assigned in the source program, A = 0.

2. B is the internal formula number (i.e., a number assigned by the Compiler to each source statement) of the source statement that contains the subroutine call.
3. C depends on the type of subroutine called:
   a. If the calling sequence appears in an arithmetic statement function definition, $C=32767_{10}$.
   b. If a. is not satisfied and the calling sequence appears in a FUNCTION or SUBROUTINE subprogram, C = \$+2.
   c. If neither a. nor b. is satisfied, C = 0.

If the call to the error program occurs in a subroutine, the NTR and PZE instructions give the necessary information to determine which source statement in the higher level subroutine called the subroutine containing the error. This technique can be used to trace through any number of subroutines back to the main program.

Logic of the Error Program

An error program may be written, using the following logic, by any installation desiring the standard error procedure.

1. The error program is called by a TSX on Index Register 4, which is generated by a CALL statement in the program containing the error; therefore, the location of the call to the error program is known.
2. Scan the calling sequence to skip over the TSX LARGi instructions to obtain the NTR and PZE instructions.
3. The decrement of the NTR and PZE instructions are, respectively, the external and internal formula numbers of the source statement producing the calling sequence.
4. If the address part of the PZE instruction is zero, this is a main program (no program name), and the back trace terminates.
5. If the address part of the PZE instruction is 32767_{10}, this is an arithmetic statement function. Accordingly, examine the instruction preceding the TSX of the calling sequence. This instruction is SXD L,4. The decrement of word L contains the 2's complement of the location of the TSX of the calling sequence for this arithmetic statement function. Return to step 2. to continue the back trace.
6. If the address part of the PZE instruction is neither 32767_{10} nor 0, it is the location of an instruction whose decrement gives the 2's complement of the location of the TSX to this particular subprogram. Furthermore, the address plus 1 of the PZE is the location of the BCD name of the subprogram. Return to step 2. to continue the back trace.
7. By scanning the TSX instructions of a FUNCTION or SUBROUTINE subprogram calling sequence, the error program can determine the current values of the subprogram arguments.

APPENDIX C: SYMBOLIC/ABSOLUTE LOCATION REFERENCES

Symbolic locations referred to in this manual are listed below with their corresponding absolute locations in octal.

		Assignment Location when Operating Under	
Symbol	FORTRAN Record	FORTRAN II Monitor	IBSYS
TAPRO	IOP	75051	not applicable
DATEBOX	SST	142	3732
(DATE)	SST	73471	73662
(PGCT)	SST	73460	73651
OPTION	Sign-on	154	not applicable
HTPSTS	Sign-on	1141	4725
FCTWO	Sign-On	177	3761
GTCOM	Sign-On	1145	4731
STAT16	Sign-On	1270	5056
CORBF	Sign-On	3312	7210
ORDBF	FAP	545	545
WRITS	FAP	703	703
SPACS	FAP	2571	2571
ORDRS	FAP	4342	4342
INT00	FAP	5276	5276
ERRRS	FAP	6024	6024
OPDOP	FAP	6025	6025
CHANS	FAP	146	146
SYSAST	FAP	147	147
SYSTPS	FAP	150	150
(ENDS)	SST	73474	73665
TOPR1	BSS	72227	72400
TOPR2	BSS	72364	72536
NOSTON	BSS	73054	73231
NOSTOF	BSS	73055	73232
(LIBT)	SST	73461	73652
BOTTOM	SST	144	3734
MEMRY	Section 6	144	3734
BOTTAB	Section 6	10270	14603
TOPTAB	SST	73377	73577
XMTST	Section 6	1531	5365
CODE	EXEM	562	315
CONTE	Gen Diag	230	4023
ONLINE	Gen Diag	1555	5350

INDEX

C28-6066-6

Printed in U.S.A. C28-6066-6

IBM

International Business Machines Corporation
Data Processing Division, 112 East Post Road, White Plains, N.Y. 10601

The instructions below show a typical transfer list and prologue.

```
SUBP1  BCD   1SUBP1
SUBP2  BCD   1SUBP2              Transfer List
  .     .      .
  .     .      .
  .     .      .
SUBPN  BCD   1SUBPN

       HTR               Storage for contents of IR4
       HTR               Storage for contents of IR2
       HTR               Storage for contents of IR1
NAME   SXD   NAME-3,4    Save IR4 contents in (NAME-3)
       SXD   NAME-2,2    Save IR2 contents in (NAME-2)
       SXD   NAME-1,1    Save IR1 contents in (NAME-1)
       CLA   1,4
       STA   X1          Location of 1st argument→X1 21-35
       CLA   2,4
       STA   X2          Location of 2nd argument→X2 21-35
       CLA   N,4
       STA   Xn          Location of nth argument→Xn 21-35
```

Results

A FUNCTION subprogram must place its (single) result in the accumulator prior to returning control to the calling program.

A SUBROUTINE subprogram must place each of its results in a storage location. (Such a subprogram need not return results.) A result represented by the nth argument of a CALL statement is stored in the location specified by the address field of location (n, 4).

Return

Transfer of control to the calling program is effected by:

1. restoring the Index Registers to their condition prior to transfer of control to the subprogram, and
2. transferring to the calling program.

The required steps are as follows:

```
LXD    NAME-3,4    RESTORE CONTENTS OF XR4
LXD    NAME-2,2    RESTORE CONTENTS OF XR2
LXD    NAME-1,1    RESTORE CONTENTS OF XR1
TRA    N+1,4       RETURN. N=NUMBER OF ARGUMENTS
```

Entry

Unlike a FORTRAN compiled subprogram, a hand-coded subprogram may have more than one entry point. A hand-coded subprogram used with a FORTRAN calling program may be entered at any desired point, provided that a subprogram name acceptable to FORTRAN is assigned to each selected entry point. All the above mentioned conditions must, of course, be satisfied at each entry point.

System Tape Subroutines

As discussed previously, hand-coded subprograms as well as Library functions may be placed on the System tape of the FORTRAN System. When a FORTRAN source program mentions the name of such a subprogram, it is handled in exactly the same way as a library function.

Alphameric Information

Hand-coded subprograms may handle alphameric information. This information is supplied as an argument of a CALL statement. The form of an alphameric argument is:

$nHx_1x_2 \ldots x_n$

The following example illustrates the method of storing alphameric information.

CALL TRMLPH (8, C, 13HFINAL RESULTS)

The characters 13H are dropped and the remaining information stored as follows:

Location	Contents
X	F I N A Lb
X+1	R E S U L T
X+2	S b b b b b (b represents a blank, 60_8)
X+3	777777777777_8

The address X is given in the calling sequence for the CALL statement.

C28-6054-5

Printed in U.S.A. C28-6054-5

IBM®

International Business Machines Corporation
Data Processing Division
112 East Post Road, White Plains, New York 10601